

S I N C L A I R

Every month £1.75 November 1989

QL WORLD

SOFTWARE FILE:

The Blag II

INSIDE FRONT COVER

4 to 9 8 9 13 14

22 circuit mobs 24 26 ROM 29

30 34 40

THE MINERVA EPROM

An operating system update
takes out some bugs

TECHNICAL HELPLINE

Crashproofing
the QL



Editor
Helen Armstrong

Chief Sub Editor
Harold Mayes MBE

Production Manager
Nick Fry

Designer
Chris Winch

Advertising Sales
Jason Newman

Magazine Services
Sheila Baker

Advertising Production
Michelle Evans

Publisher
Perry Trevers

Managing Director
Peter Welham

Financial Director
Brendan McGrath

Chief Executive
Richard Hease

Microdrive Exchange
089 283 4783/2952
(2 lines) TIL

Sinclair QL World
Greencoat House
Francis Street
London SW1 1DG
Telephone 01-834 1717
Fax 01-828 0270
Telex 9419564 FOCUS G
ISSN 026806X

Unfortunately, we are no longer able to answer enquiries made by telephone. If you have any comments or difficulties, please write to The Editor, Open Channel, Trouble Shooter, or Pison Solutions. We will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.

Back issues are available from the publisher price £2 U.K., £2.75 Europe. Overseas rates on request. Please telephone 089 283 4783 to check availability.

Published by Focus Magazine Ltd., London, 5 M Distribution, Sreatham, London SW1. 01 877 8111. Subscription information from: TIL, PO Box 74, Paddock Wood, Tonbridge, Kent TN12 6DW. £21.00 U.K., £24.70 Europe, Middle East £25.80, Far East £27.60, Rest of World £28.20, U.S.A. \$45.00. Airmail rates available on request 0893 534783. Typesetting by Adtec Typographics, Stanford-le-Hope, Essex. Tel: (0375) 560967.

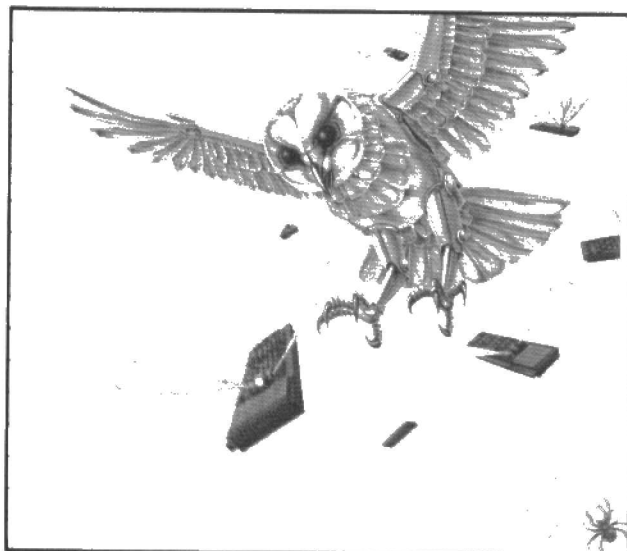
Printing by Southernprint Ltd. Sinclair QL World is published on the fourth Wednesday preceding cover date.

©COPYRIGHT SINCLAIR QL WORLD — 1989.

CONTENTS

■ ■ NOVEMBER 1989 ●

- 9 **QL SCENE ● Minerva offer to club members**
- 10 **OPEN CHANNEL ● Ones and zeros**
- 13 **QL SCENE ● A new Solution**
- 14 **TECHNICAL HELPLINE ● Confusing questions answered**
- 17 **SOFTWARE FILE ● QZ QL/Z88 file transfer**
- 18 **ONE MAN'S SYSTEM ● Praise the QL**
- 22 **CRASHPROOFING THE QL ● A digest of suggestions**
- 26 **MINERVA – THE ROM ● A replacement ROM for the QL**
- 30 **ARCHIVE SCREEN DRIVER ● Another angle on Archive**
- 33 **SOFTWARE FILE ● The Blag 2**
- 34 **SUPERBASIC ● More file management**
- 44 **THE PROGS ● Jupiter, Pip, BinQL**
- 46 **SUBSCRIPTION INFORMATION**
- 48 **MICRODRIVE EXCHANGE ● It's Cricket**



**NEXT
MONTH
QL ARTIST OF THE
YEAR
COMPETITION 1989**

This is getting closer all the time.

**YOU CANNOT
MEAN . . .**

Super Toolkit II from the bottom.

QL

SCENE

PDQL ready to launch

Three major new productions from PDQL are now ready for release. *Hardback* and *Finder*, for hard disc users, have been held back waiting for both the Miracle Systems and Rebel hard disc software to reach a mutually stable state. *Hardback* is a multi-tasking file management controller for hard disc, allowing Total Save, Since (the last) Save, by Directory and Save by Marking. It can deal with files of a size exceeding one or two floppy discs, as well as zero-length files.

Features also include flexible windows, file deletion, viewing, saving and restoring, directory printouts, file details display and file or file-content string search. The package is fully Thor-compatible and costs £35.

SuperBasic C-Port is the improved version of Basic C-Port re-written to handle structured and unstructured SuperBasic source files and ready for ANSI, lattice or PDQ-C compilation.

The program is a compiler, translating from SuperBasic to C. The C-Ported source file can be compiled for more effective use of the QL or, alternatively, DiscOvered for compilation in an IBM, CPM or BBC environment.

SuperBasic C-Port costs £79, with additional toolkit modules for Toolkit II or Turbo available at £30 each.

PDQ-C is a fast C compiler comparable with the Atari Lattice C. The program has a large, tested and documented library and is designed to handle any size of program subject to memory restriction. PDQ-C costs £79.

Orders and enquiries to PDQL, 1 Heaton House, Camden Street, Birmingham B1 3BZ.

QView will hold Minerva payments

Minerva is the 'new and improved' QL operating system from QView. Copies will be supplied for £30 to users – £25 to current Quanta or QLSub members – who can send a microcassette or disc – 3.5in. or 5.25in. – containing an image of their original QL ROM. That is to ensure that customers already possess a copy of Qdos and are entitled to use it. *Minerva* then acts as an update to Qdos. *Minerva* is supplied on EPROM. Microdrives and discs will be returned with any information on subsequent changes in *Minerva*.

QView is advertising non-solderable installation – the ROM rests in an IC socket – faster TRAP entry, faster scheduler, faster floating point arithmetic, faster string manipulation and concatenation, faster program search – GOTO, PROC call and so on – a major rewrite giving faster and more accurate graphics, faster RAM test for Trump Card owners, second screen via

a new MODE command and TRAP, COMPOSE foreign characters and extended character set – including Greek – SuperBasic TRACE hooks including single step, upside-down SCALE, ATAN (x,y) to ATAN (y,x) faster DATE procedures with more flexible parameters, implementation of string and integer SELECTS, F1/F2 auto-start after 30 second timeout, F3 restart with second screen enabled, trapping of line 1010 and 1111 emulator exceptions, extra task and screen switching keys, warm reset via CALL, extended plug-in ROM scanning, ESCape in EDLINE (AUTO and EDIT), rapid execution of multiple QLib jobs, extra RI functions from machine code RI.EXE, PAUSE takes channel number like INKEY\$, BASIC RESPR utilising Common Heap when jobs are running 8-pixel wide fonts, STRIP size to match character size, ABS(n1,n2,...) giving SQRT(n1²+n2²+...

...), WHEN error and WHEN variable now functioning, maximum use made of INTEGER arithmetic, a\$(TO 5) now defaulting to a\$(1 TO 5), integer and string FOR/END FOR loops, VER\$ giving Qdos version and system variable base, and graphics equalling 92 per cent of Digital Precision *Lightning* as measured with DP DEMO_GRAF_BAS.

There are also various bug fixes to the original ROM. A full review of *Minerva* appears on page 26 of this issue of *QL World*.

Orders and enquiries to QView, 29 Carnaby Close, Godmanchester, Cambridgeshire PE18 8EE. Tel: 0480 412884. Quanta and QLSub members should send a copy of the current newsletter envelope to claim the £5 discount. Delivery is projected for seven days from receipt of order and QView states that cheques will in any case not be cashed until despatch.

Lightning strikes twice

The famous Digital Precision speed-up program *Lightning* has spawned an even brisker offspring, *Rom Lightning*. As part of a special edition release of *Lightning*, *Rom Lightning* is typically 30 percent faster than its predecessor, according to DP tests.

DP states that running the program from a plug-in ROM has given it more elbow room and allowed them to concentrate entirely on speed. The ROM contains the automatic

routines for text and screen acceleration. The maths and graphics enhancements need to be loaded from disc or microcassette if required.

There are a number of new features including, strikingly, a utility to allow all the channel characteristics of any task running on the QL to be user-adjusted at the time of running. Normally the user should need only to fine-tune the font and ink/paper/strip colours but the adventurous can adjust vertical and horizontal character spacing, vertical and horizontal

character size, and window shape, size and position. There is also a scroll speed control which can be adjusted from very fast to a smooth one-pixel-row at a time scroll.

The new manual is considerably enlarged.

Rom Lightning Special Edition costs £49.95 from Digital Precision Ltd, 222 The Avenue, London E4 9SA. Tel: 01-527 5493. Purchasers should specify ROM plus disc, or ROM plus cartridge. Current *Lightning* users can upgrade.

OPEN CHANNEL

Open Channel is where you have the opportunity to voice your opinions in *Sinclair QL World*. Whether you want to ask for help with a technical problem, provide somebody

with the answer, or just sound off about something which bothers you, write to: Open Channel, Sinclair QL World, Greencoat House, Francis Street, London SW1 1DG.

Front Page

Having attempted to obtain a copy of *Front Page* (128K) for my QL from TK Computerware and from every other known software house and even attempted to trace the program originator, without success, is anyone on the QL scene willing to let me have a copy of *Front Page*? I am afraid expansion is out of my price range for *Front Page Extra*. Any expenses incurred would be met.

R. Bristow,
12, Ashurst Gardens,
Cliftonville,
Kent CT9 3HW.

Oh, oh

Please discourage people from using ambiguous letters when submitting programs for publi-

cation. I refer in particular to i's and l's and O's and 0's.

Which of the last two is a zero? Which of the previous two are letters or numerals? So often to find the answer requires close study of the program, wasting valuable time.

This unnecessary use of ambiguous letters in a program is not only difficult for amateur typists but must be equally difficult for typists and typesetters.

Eric Starling,
Largs,
Ayrshire.

Editor's reply: Ones and Is, zeros and Os are usually distinctive in the QL World two typefaces. Zeros should be narrower (0) than Os and ones (1) have a longer serif than Is. Listings attached to programs

in The Progs frequently are printed on the author's own printers, as we try to avoid re-typing listings wherever possible. Occasionally a typeface does not distinguish between the pairs, especially ones and Is.

Programmers should avoid using 0 or 1 in positions where another character would serve and be careful not to type letters for figures or vice versa when describing programs.

E bug

Two points arising from the August issue. I enjoyed Mike Lloyd's article about Super-Basic with its attendant 12 listings but there is a bug in listing seven. Run this:

```
100 CLS: INPUT word$
110 x = "0" & word$
120 PRINT x
```

It will print zero as expected unless the input begins with the letter 'E', when it will crash. I discovered this by accident and I believe it has something to do with exponential notation.

My second point concerns Martin Wheatley's letter about fitting Brother paper into the SER 8056 printer. I have the same problem but I do not understand his solution involving spring clips from TV aerial adapters. What does he mean and can he elucidate?

C.B. Storey,
Whitley Bay,
Tyne and Wear.

Mike Lloyd replies: You are correct about the 'bug' and about its cause. You cannot really call it a bug, because the Qdos mathematics suite is designed to recognise an 'e' in numbers. The problem cannot be avoided as 'e' is the most commonly-used letter in English words. Programmers will have to precede the error trap with another

routine to prevent the symptoms you report.

*Editor's comment: Some readers have noticed that the formula correction published in last month's Open Channel still lacks one thing – the closing bracket between the first 12 and the * which follows it. LLoyd sent two copies of the correction, one for QL World and one for enquirer Eric Sargeant. We kept the one with the mistake in it. How easy it is.*

Best cut

I spoke to Captain Norman Vasher of Gulf Air who cut off the Microdrive end of the QL and made it work. Jones of the sawn-off QL wrote to me about cutting the end off his QL and I referred him to Vasher.

Vasher states that there are only two or three lines which need hard wiring and a transistor which needs adding. What you lose in the conversion is the Microdrives, network, TV output and so forth, but all the lines are available. They amount only to four.

What you get if you use a standard computer supply of +5V, +12V and -12V is a much better display, better reliability, cooler running, no crashes, and so on. Obviously none of your experts had tried it and the non-expert who tried it had succeeded.

Dennis Briggs,
Adman Services,
Telford.

Editor's reply: We thought it might be a belated April Fool joke, particularly as no address was included in the letter. It was Jones' tumultuous approach, rather than his aspirations, which really alarmed us. There are at least two people on our experts' committee who would rise to truncating the QL if an

Editor's notebook

First, apologies for the non-appearance of the Artist of the Year 1989 Competition this month. It is in progress but one of the main organisers has been engaged in productive work and was unable to meet the deadline. Next month, we hope...

Digital Precision has announced the first major upgrade of the *Solution* and the curious are queuing to review it. We shall be speaking to Freddie Vachha when he returns from Belgium, or soon after. Meanwhile, no further news of Transformer has been received in recent weeks.

The West Midlands Quanta Group, based at the Holloway pub in Birmingham is getting bigger all the time and limited in numbers, I am told, only by the size of the room. It has just celebrated its fifth birthday. Happy birthday, everyone.

We have not yet found Mr. Parrott but another victim of the Great Postal Packet Disaster returned recently from overseas and found our letter – six months later. Even panic seems unrewarding at this stage. Send tranquilisers for the attention of the editor, please.

urgent need arose but not, I think, without a written disclaimer from the owner. *QL World* decided that it would not take that initiative. After all, even Briggs does not mention 'being less wide' as a major benefit.

I am very pleased to hear that Jones has found the advice he needs and wish him a long – or shorter – and happy relationship with his new QL.

Worst cut

With reference to Jones' dilemma, he should try incubating the segments of his QL in a bed of chopped WORMs. With luck, they would regenerate as complete QLs, with interesting memories.

C.R. Oswin,
Christchurch.

Index

Anyone tired of skimming through back issues of *QL World* and *QL User* looking for information might be interested in an Archive-based index program I have put in the Quanta library.

It covers all *QL World* issues to September and *QL User* from mid-1985 to its merger with *QLW*. There are 1,900 records on one disc or two Microdrive cartridges. You will need memory expansion to run it. There is also a program to update the database month by month, though I hope to keep the library version up-to-date.

Christopher Adams,
Moseley,
Birmingham.

VDUs

I wonder if any readers might suggest a simple way of hanging a VDU on a QL? I use my machine for accounts work and need the use of a numeric keypad which is lacking on the basic QL. My solution has been to obtain a second-hand VDU, which cost next to nothing, complete with a reasonably robust keyboard and to plug it into the SER port.

Now the problem. I need to write some code to handle the data coming from the VDU. With the terminal in its half-

duplex mode, all keyboard depressions are displayed faithfully on the VDU and data is squirted to the QL when you press ENTER or similar.

An INPUT in\$ command will receive this data but you still have to filter out any cursor characters, such as backspaces, which may have been introduced. If you turn the VDU into its full-duplex mode you can use the INKEY\$ command to receive the data character by character but you then have to send back the data to the VDU screen to display it.

I feel sure that there must be a simple way of telling the second processor in the QL to transfer its affections to another keyboard. If so, users could obtain relatively cheap old VDUs with proper keyboards and we could all leave the standard "plastic plugs" one alone.

David Spratt,
Horncastle,
Lincolnshire.

Pipes

May I sound a word of caution on pipes? For a simple system with one or two pipes and only one or two concurrently-executing programs, there is little likelihood of trouble. If you have several pipes and co-executing programs, sooner or later disaster will strike.

With any filestore device, such as a Microdrive, when you try to read and write to the same file simultaneously you will get an 'in use' error. That error does not occur with a pipe, so two co-executing programs can read from and write to the same pipe as near simultaneously as Qdos will allow. At best this will lead to lost or corrupted data; at worst to a system crash.

Ideally, one would write a pair of pipe drivers for input and output pipes which could be numbered so that the drivers can link both ends and which would include a pipe lock. As we do not live in an ideal world I use a block of memory – one byte per pipe – to form a pipe lock. The address of the block is passed to all the programs which use the pipes. The lock byte for each pipe is allowed to hold only one of three values; 2, 1 or 0. 2 indicates that the input pipe is open, so the output pipe may now be opened; 1 indicates that the pipe has been written to and a read is awaited/in progress; 0 indicates that the pipe

has been read and a write is awaited/in progress. The byte lock value is changed only on completion of the relevant action and the action is allowed only if the lock byte holds the appropriate value.

This pipe locking method is not very elegant but it has two major attributes – it is simple and it works.

For Prospero Pro-Fortran 77 users like myself the channel number of the input pipe can be passed in an adjacent block of memory but the Prospero peek/poke must be used with caution. The manual is not clear but the value to be peeked or poked must be a form-byte integer. You poke the low byte only at the stated address and peek the byte at the address into the high byte of your variable.

The lower three bytes are not zero. That top byte must then be extracted, remembering that it may be positive, negative or equal 128. A simple divide by 16777216 may not suffice. That apart, Prospero F77 is far and away the best and most accurate implementation of Fortran 77 I have seen.

As a final comment, now that we have hard discs, when will someone port Unix on to the QL and give us access to an even greater source of software?

G. S. Worsnop,
Sutton,
Surrey.

Oki help

Can anyone help me with the Oki Microline 292 printer with the QL? I have rewritten the Quill printer driver and get reasonable results using continuous paper. The problems start when I use the cut sheet feeder; the first sheet is fed correctly to TOF but subsequent sheets are fed incorrectly. If, from SuperBasic, I send to the printer the ASCII code FFchr\$(12)), it feeds correctly to TOF every time. If I use the printer front panel form feed button, again it feeds correctly every time.

My second problem concerns screen dumping to the Oki. I have a Trump Card. The Trump SDUMP facility seems to crash the printer. The Oki requires ASCII ETX (chr\$(3)) to go into graphics mode. The Trump Card sends ASCII ESC = (chr\$(27);chr\$(42)). Miracle Systems can offer no help. Does

anyone have a screen dump program for the Oki?

Finally, I am amazed that the QL is still going strong, with so much hardware and software still available, and people like Miracle Systems spending so much time and effort to develop new products.

I bought my QL almost exactly four years ago. For three years it had very little use, as it was a basic system running on a black and white television set. My situation changed financially, I brought the Trump Card and the twin 3.5in. drives and now with the printer the system is an excellent tool, not just a toy. Keep up the good work.

M. J. Simms,
56 Mitchelmore Road,
Yeovil,
Somerset BA21 4BA.

Graphs

Using a Citizen or any Epson-compatible printer, by modifying the Easel screen dump GPRINT__prt file, I can now print four graphs on one A4 sheet of paper. First place a back-up copy of Easel in mdv1__1, then type-in the following commands:

```
A=RESPR(600)
LBYTES MDV1__GPRINT__
PRT,A
POKE A+410,6
POKE A+413,90
SBYTES MDV1__SMALL__
PRT,A,512
```

Load MDV1__BOOT,
RENUM and enter the following lines:

```
101 OPEN#5,SER1
102 INPUT 'ENTER COLUMN
NUMBER (0 TO 40) YOU
WANT TO START YOUR
GRAPH';C
103 PRINT#5, CHR$(27);
'1';CHR$(C)
104 CLOSE#5
```

```
DELETE MDV1__BOOT
SAVE MDV1__BOOT
LRUN MDV1__BOOT
```

Note: using column 0 or 409 you will be able to print two graphs side by side. When using Easel, press F3, (P)RINT, (I)NSTALL MDV1__SMALL__PRT. Before running Easel, make sure your printer is switched on to accept print codes from the boot.

S. T. Cresswell,
Heanor,
Derbyshire.

QL

S C E N E

The final Solution?

Art from Belgium

At the end of 1988 Digital Precision released the PC emulator *Solution*, which has received favourable reviews from a number of sources including *QL World*. The primary reservation expressed by testers was lack of speed in many functions, although in out-performed emulators on other well-known computers in many ways.

Now the DP team led by Steve Sutton, the author of *Lightning* and many other DP programs and an expert in speeding things, has produced breakthroughs which accelerated PC emulation on the QL.

The result is a new program, *PC Conqueror*. *Conqueror* will run almost twice as fast as *Solution* on a wide range of PC programs. DP technical director and supremo Freddy Vaccha is amazed at the increase and would like all *Solution* owners to upgrade 'quoting the following test:

Enter a FOR ...NEXT loop in Microsoft Basic and time it; *Conqueror* will clock in at 85 percent faster than *Solution*. Disc formatting is quoted as 50 percent faster, and boot time as low as 30 seconds. The jerki-

ness frequently experienced while typing with *Solution* has been eliminated.

Additionally, *Conqueror* is more widely compatible than *Solution*, able to handle programs which grab the keyboard, and designed to be compatible with the Thor, Miracle and Rebel hard disc systems. More compact than *Solution*, *Conqueror* incorporates some half-dozen new options.

Conqueror costs £89.95 with

an additional £50 if MS-DOS V4.01/GW-Basic is required as well, prices only slightly above the original 'vanilla' and 'chocolate' versions of *solution*. *Solution* had a substantial price reduction slash to £39.95. Users upgrading from *Solution* are promised prices "substantially less" than the quoted price for *PC Conqueror*.

Digital Precision, 222 The Avenue, London E4 9SE. Tel. 01-527 5493.

Accounts from S.D.

S D Microsystems, which specialises in small business software for the QL, including the *Small Traders' Pack* and *General Ledger*, has now published the *Stock Accounting System*, an integrated office package.

Unlike the two earlier packages, *Stock Accounting System* requires at least 256K of expansion and combines invoice production with stock control and accounting. Invoices are prepared with a built-in product/price table, with stock level adjustment and sales ledger posting performed automatically.

All files are loaded in one go with no dependence on overlays from disc. Up to 999 lines of stock are available on-line instantly. The standard version of the program produces invoices, statements and credit notes on plain or letter-headed paper. An option will be offered which uses specially-produced NCR invoice stationery designed by a leading PC software house for a fully professional appearance.

Orders and enquiries to S D Microsystems, PO BOX 24, Hitchin, Herts. Tel: 8462 675186.

Joachim and Nathan Van der Auwera, authors and publishers of *The Painter*, have contacted *QL World* to inform us that *The Painter* has been updated since our recent review - *QL World*, July, 1989 - with certain errors corrected and most routines, particularly screen-building, speeded.

The Van der Auweras are now also marketing *The ClipART*, which consists of three discs and 150 screens of artwork. The *ClipART* is priced, perhaps somewhat disconcertingly for buyers outside Belgium, at 2,150 Belgian francs, inclusive of postage and packing. To obtain the update to *The Painter*, send the original disc and 150 Belgian francs. *The Painter* is marketed in the U.K. by Schoen PCP but it is not known whether Schoen has *The ClipART* and updates available at present.

The Van der Auweras can be contacted at PB 238, 3000 Leuven 1, Belgium. Tel: +32 16 48 89 52.

Tape found

Ablex, manufacturer of the QL microcassettes, report, that it has sourced replacement tape for microcassette production successfully and will "have stocks to see us well into next year", according to production manager David MacSorley.

Earlier this year Ablex told *QL World* that it would continue to manufacture microcassettes to the end of 1989. Observers now believe that it will continue to meet demand while it is economic and tape supplies last but will be unlikely to source further tape if another shortage occurs.

Alternative Micro Show

The Third Alternative Micro Show and Electronics Fair will be held at the Bingley Hall Staffordshire Show Centre, Stafford on Saturday, November 11 from 10am to 5pm.

The Alternative Micro Show is making a name as the primary show for users of specialist or non-mainstream micros, bringing the smaller markets together at a major venue to

attract a greater number of general and peripheral suppliers. The show is widening its frame of reference still further this year, featuring as previously Einstein, Dragon, MSX machines, Lynx, Texas TI, Oric, Jupiter Ace and Enterprise, but adding to the roster "all the other micros except the ST/Amiga/PCs.

For the first time this year

the show will also include an electronics fair for home builders and designers. There will be a bring-and-buy sale in the fairground tradition.

For further information contact the organiser, Taurus Computer Systems, Enterprise House, Unit 15, Riverside Industrial Park, Rapier Street, Ipswich, Suffolk IP2 8JX. Tel: 0473 602460.

TECHNICAL HELPLINE

Dennis Briggs "and friends" take on another batch of technical queries from the Helpline sack.

Basic book

Having just acquired a QL with no frills or expansion I am keen to learn about it. I wonder if there is a book which will save me time. Perhaps you can recommend some? Is there a QL group in my area?

I believe I have the JM version of the ROM. Is there any real advantage in getting a JS ROM?

Brian Hulatt,
Sunbury-on-Thames.

Quanta has recently published *QL SuperBasic - The Definitive Guide* by Jan Jones, costing £8 plus £2 post and packing. It is widely considered to be exactly what its title suggests. There is also at the time of writing a local Quanta group at Sunbury-on-Thames. Contact Quanta, c/o Phil Borman, 15 Grosvenor Crescent, Grimsby, South Humberside DN32 0QJ for more information.

To check your ROM, power up the QL, select F1 or F2 and type:

PRINT VER\$

and it should print the ROM version in the top left-hand corner. There are advantages in having the JS ROM in that some programs, particularly updates of programs which were around when the JM ROMs were issued, do not run properly on JM machines. It is probably better, in the first place, to enquire about this when buying software and see how you fare than to buy a new ROM but it also depends on how much you have to spend. A new ROM costs around £30, as does an updated set of the Psion quartet. TK Computerware - 0303 81 2801 - can help on both counts.

Dead tube

I have the Sinclair Vision QL colour monitor with a 12in. screen. It is made by Kaga

Electronics Co Ltd of Japan and was sold by Dixons in large numbers.

The tube on it appears to have expired. How can I obtain a new cathode ray tube and get it fitted?

William Hutton,
Barmouth,
Gwynedd.

Adman Services has circuit diagrams for the monitor and can repair it. The telephone number is 0952 255895.

C compiler

I would be grateful if you could recommend me a C compiler. I have seen various ones advertised in the magazine but wish to know which has the fullest implementation, particularly regard to structures.

Ian Jackson,
Pocklington,
E. Yorkshire.

The Metacomoco 'C' compiler has been updated recently by Chas Dillon with the co-operation of Metacomco. It is available from PDQL in Birmingham, which is the distributor for Dillon's software and responsible for all after-sales support. This version of the 'C' compiler has none of the shortcomings of the previous compilers and should meet your needs admirably.

Look at the full range of C-related programs available from PDQL as there is a special potentially wide-ranging one which takes SuperBasic source files and turns them into 'C' source files ready for compilation.

Sync hitch

I have a Texas 820 Keyboard Send Receive terminal printer connected to the QL SER1 port which provides high-speed program listings. It would be

useful to use the keyboard together with or instead of the QL keyboard to control the QL and for input to word processor programs such as Quill.

I have read the QL Advanced User Guide and the Sinclair QDos Companion, which have provided some indication on the use of channels (TRAP #2) and serial I/O operations (TRAP #3). Apart from the TRAP #1, MT_IPCOM function, there seems to be little information on how or where the QL keyboard input characters are received, stored and used to control the various Qdos and SuperBasic functions, which might allow an independent machine code programs to link in characters received from the Texas 820 keyboard via SER1 as an alternative controlling device and still provide a program listing facility as well.

Can this facility be provided by a relatively small assembler program? If the answer is yes, how can it be achieved? Would this facility be useful to other users who may have access to a serial keyboard terminal device?

Len Watts,
Rainham,
Gillingham.

There has always been a demand for alternative keyboards for the QL, mainly for personal preference. Schoen has two types, both of which are available. One is a simple add-on using the existing keyboard connections but needing a different co-processor chip to prevent double-strike of the keys. Unfortunately it can introduce a side effect on some QLs which prevents the serial ports functioning correctly. The same company also supplies a PC-style keyboard using a different approach.

A special type of PC keyboard can be hooked to a QL by using the ABC Electronics interface. Both PC-type

keyboards cost about £100 with the necessary interface.

If you are willing to forego the use of the serial 2 port, any serial keyboard with an RS232 output can be used. The program should make matters clear. Notice that it does not matter what is sending the data to the serial port. It could be a keyboard, modem, Microwriter or another computer. It just changes what goes in to what makes sense to the QL.

The TRANslate function of the JS and MG ROMs are a similar type of look-up table.

RS to keyboard translator. This is a routine to take characters entering through SER2 and place them in the keyboard queue. It therefore makes it possible to connect any program needing a keyboard input direct to the outside world. Uses include an external keyboard such as a terminal or Microwriter, optical scanner, modem direct into Editor, and even a PC mouse.

Qontrol

We have had a Sinclair QL in the family since early 1985 and it is still working impeccably every day. In recent years we have used it together with the Phillips monochrome Monitor 80. We have just had the opportunity to buy a Nixdorf BA23 colour monitor at a bargain price.

We have the necessary pin-out information for the QL and the monitor. There is a hitch, though; the monitor requires a horizontal and a vertical sync signal while the QL seems to offer only a composite sync.

Is there a way to break down the composite QL signal into its horizontal and vertical components? Or is there any other way round the problem? I would appreciate your shedding some light on the situation.

Gunnar Oehr,
Farjstaden,
Sweden.

The QL monitor connection supplies vertical and composite sync as well as RGB, composite video and composite mono. On many monitors the composite sync will be adequate for horizontal sync purposes. If the signal needs separating the enclosed circuit will do the trick. It is just a simple sync separator. It is probable that the line sync pulse which is a positive one from the QL will need inverting to get your monitor to function. A single transistor can be made to invert the signal or just use one logic gate to do the trick.

Q-Connect

I have a Spem QL casing and have fitted Miracle Expanderam in the lower slot of the bus extension board. I have also fitted a CST disc drive interface in the top slot, a combination which works well. My problem is that I have a Qontrol interface board I need to use in conjunction with the disc interface as I wish to run my video digitiser board from the Qontrol board.

The video board is a kit from Maplin Electronics which needs two TTL ports to control it, the problem being that the Qontrol board needs to be in the top slot of the Spem extension, which means that the disc interface cannot be fitted. If it is fitted on the through connector on the Spem extension all I get when the system is switched on is a white screen with no messages. Could you tell me how I can fit all three boards into the system so that they all work together?

**M.R. Howman,
Bourne,
Lincolnshire.**

I can only presume that you are trying to use the Colin Opie control board along with the Expanderam and the CST disc interface. As I have no details of the control board I can only assume that there is a conflict where all the boards are in the address space.

In effect, one appears to be stamping all over another, hence the white screen. If you could supply more details I can look at the problem in greater depth.

Eventually I found with great difficulty the answer to your problem. Fit the Colin Opie board to the QL and the Expanderam into this board,

not the other way round. The reason is that the Expanderam carries only the few lines needed for the disc interface. Many lines are not carried through.

On many European televisions there is a SCART socket with RGB and sync inputs. The pin-outs are enclosed with the RGB pins highlighted. Just solder the QL monitor lead as shown and it will work.

Mystery

I have all three modules of the Tandata Q-Connect modem which I find works extremely well on Microdrive cartridge, except that it takes a very long time to load. Can you tell me how to adapt it for a Cumana disc drive? I have tried adding FLP_USE MDV into the boot program but it refuses to accept this command at line 150, for no apparent reason.

**Terry Gould,
Bugbrooke,
Northampton.**

The simple answer to using the Tandata modem software is either to compile it with Q-Liberator or at least use Q-Liberator Q-LOAD after it has been Q-MAVED. This piece of software takes a snapshot of the program in RAM, saves it to Microdrive or disc when Q-Loaded, loads it back in exactly the same position ready to run. It reduces loading by a substantial amount but, of course, it cannot change the speed at which data is transferred from tape.

Disc drives are very much faster in transferring data across, therefore investment in them is worthwhile. If this is not a suitable path for you I suggest you buy the QualSoft terminal software from TF Services, 12 Bouverie Place, London W3.

I have been unable to use my QL for more than a year. Suddenly it refused to print a document from Quill. When I try to print I get "Unable to open file. Error - press space to continue." I tried copying a fresh tape from my Master Quill 2.35, using the attached listing to install my Brother M1009 printer. Nothing worked. Prior to this fault I had used my QL for two years without difficulty.

I put in my QL for repair but on having it returned I still could not print a file. Therefore I put in the QL for repair again. I was told the QL was in perfect working order. Fortunately, or unfortunately, it was lost in the post. I was very lucky to have it replaced by a new QL. On returning home, I found that the new QL did not work but had the same fault.

Therefore I put in my printer for repair. My printer was returned with a test printout showing that it was in perfect working order. Therefore I was left with my QL and printer having been tested and found to be in working order but I still cannot print a file. So after considerable cost and a year without my QL I am no further forward. Every function works except the print command.

**Dr. N.P. Halliday
Guildford,
Surrey.**

As you have had your QL replaced and had the printer tested, the only other component in the chain is the printer lead. I have just encountered another QL owner with the same set-up as yours and the same fault. It proved to be the printer lead.

Adman Services, 53 Gilpin Road, Telford, Shropshire can supply you with a test cable for the QL which connects serial port 1 to serial port 2. You send data in the form of characters out through serial 1 and receive them through serial, then reverse the process. This proves that the serial ports are functioning correctly. The following basic does the complete test:

```
100 OPEN 3,ser1: REMark
Change to ser2 for reverse test
110 OPEN_IN 4,ser2:
REMark Change to ser1 for
reverse test
120 REPEAT loop1
130 for a=0 to 255
140 print '3,chr$(a);
150 PRINT inkey$('4,(-1));
160 end for a
170 END REPEAT loop1
```

The same supplier can also supply a new printer lead for £8.50 and has a small monitor which plugs between the QL and the printer lead to show the signals present on the serial cable by a system of coloured lights.

There are two important points

to remember when using this combination. One is that you will have to install the printer on Quill and Abacus by typing `LRUN MDV1_INSTALL_BAS` and then follow the screens to enter the data listed below.

The second point concerns Easel and Archive as those packages cannot be reconfigured. It is for that reason that the baud rate should be set to 9,600 which has so far prove correct for other packages.

Parity	Even
Baud rate	9600
Preamble	27,64,27,82,2
Emphasised	on 27,69
Emphasised	off 27,70
Underline on	27,45,1
Underline off	27,45,0
Subscript on	27,83,1

Subscript off	27,84
Superscript on	27,83,0
Superscript off	27,84
Translate	96,35

Printer (used with Quill or Abacus):

Switch 1

Pin 8	Off
Pin 7	On
Pin 6	Off
Pin 5	On
Pin 4	On
Pin 3	On
Pin 2	Off
Pin 1	Off

Switch 2

Pin 8	On
Pin 7	Off
Pin 6	Off
Pin 5	Off
Pin 4	Off
Pin 3	Off
Pin 2	On
Pin 1	On

Printer (used with Easel or Archive):

Switch 1

Pin 8	Off
Pin 7	Off
Pin 6	On
Pin 5	On
Pin 4	On
Pin 3	On
Pin 2	On
Pin 1	On

Switch 2

Pin 8	On
Pin 7	Off
Pin 6	Off
Pin 5	Off
Pin 4	On
Pin 3	Off
Pin 2	Off
Pin 1	On

SOFTWARE FILE

QZ/QL to 288 file transfer

Last year the Cambridge Computer Z-88 was the most popular portable computer on the market and it is set to stay in the top three, despite much-improved opposition this year. It is about the size of an A4 sketchpad and less than one kilogram in weight. Recent price reductions have been made in response to competitors' price reductions and the Z-88 is finding new customers, many from the QL community.

The most obvious connection between the QL and the Z-88 is Sir Clive Sinclair, who sponsored them both. His flair for miniaturisation without compromising functionality or raising costs features in both machines, however, the lessons learned from the premature launch of the QL have shaped the development of the Z-88. The hardware has been reliable from the start and there are no obvious holes in the complex software. A surprisingly large number of QL owners have purchased the Z-88, perhaps succumbing to the power of the Sinclair name, perhaps recognising a valuable extension of their computer systems.

Members

The Z-88 is complete with an impressive array of software including a very powerful combined word processor, spreadsheet and database, *PipeDream*. Additional utilities include a diary manager, alarm clock, a structured Basic and communications facilities which allow the Z-88 to transfer data to and from other computers. Its only weakness is that, without an additional Eprom card, all programs are saved to a partition in the Z-88 RAM. Without a steady trickle of power from the computer's four AA-sized batteries the

RAM contents would swiftly fade away.

Terminal

Communications are very important to the Z-88, because it is not designed to be used as a first computer. Instead, it works best in conjunction with a host computer – a desk-top printer. The Z-88 can be regarded almost as a cable-less terminal, capable of being used many miles from the nearest power socket, with data transferable to the host computer on its return to base.

A communications program must perform two distinct functions to meet the normal requirements. First, it must be able to copy files from the Z-88 to the host computer and back again and, second, data files produced by the Z-88 software must be transferred where necessary into the format required by the programs running on the host computer.

There are three reasons for transferring data from the Z-88 to a host computer such as the QL; the first is to protect valuable data by making a backup on a disc or Microdrive attached to the QL. The vulnerable Z-88 RAM discs make regular backing-up to a more secure medium essential.

Z-88 owners may also prefer to work on a desk-top computer with a conventional keyboard and monitor display when one is available and use the Z-88 only when necessary. The rubber keys and small screen display are not the most user-friendly devices in the computer market. To swap data from Z-88 programs to their equivalents on the host computer, control codes may need to be translated and occasionally the data format needs to be modified.

Documents completed on the Z-88 might find their way to another computer to be incorporated in programs not avail-

able on the lap-top, or printed via a printer connected to the host. During the summer a great deal of my writing was accomplished on the Z-88 in the garden prior to being transferred to *Quill* for final finishing-off and printing.

David Batty's excellent little file transfer utility *QZ* does its job without fuss or bother in a reliable manner. The original version of *QZ* was rushed out soon after the release of the Z-88 and the haste of its preparation showed in the poor screen display and general lack of facilities.

Significantly, though, even this version of the software performed faultlessly.

The original user manual promised a free upgrade to the second version of *QZ* and recently I received my copy. Version two is a great improvement and has been a pleasure to use.

Settings

The main menu gives single-key access to all the program functions. When first using the program the most important option is the panel which displays the settings in force when the utility is booted. Settings can be altered and then saved so that they need not be changed again when the program is re-loaded.

The defaults include the baud rates for incoming data, outgoing data and communication with the printer, the default storage device on the QL, the serial port through which contact to the Z-88 is made and whether or not data transfer is echoed to the screen. This latter feature is useful for checking the contents of a particular file but it is as interesting as watching paint drying; the process of data transfer is accelerated significantly by opting not to echo to the screen.

Two unexpected but valu-

able defaults allow users to declare a common filename prefix and suffix which will be shared by all files received by the QL. You may wish to add the suffix "z88" to all Z-88 files, for example, or prefix back-up copies of files with the directory name "bkp".

Connection

The program is bundled with 3ft. of RS232 cable with a telephone-style socket at one end and an Atari 9-pin D socket at the other. The body of the D socket has been trimmed slightly to permit a slightly precarious connection with the Z-88 connector on the right side of the body. The new-style telephone plug fits into the ser2 port at the rear of the QL. This arrangement leaves ser1 free for the printer cable.

Let us take as a typical transfer operation the backing-up of every file in the Z-88 to a disc via the QL. The linking cable can be connected with the machines powered up or turned off. The Z for Q program runs as a task which can fit easily into a non-expandable QL. The Z-88 has its own built-in file transfer utility which is booted by pressing CTRL-X.

One thing which can be confusing initially is the need to control two computers simultaneously. Frequently I find myself typing at the Z-88 keyboard and looking at the QL screen for results. It is best to set the receiving computer into the correct mode before trying to send data from the other machine; in this example the "B" (for "Batch receive") option is chosen for the *QZ* menu.

Turning to the Z-88, its file transfer utility is booted by pressing CTRL-X. The send mode is chosen by pressing the S key, at which point the Z-88 prompts the user for a file-

name. The Z-88 operating system has a pattern-matching wildcard system similar to that found on MS-DOS and Unix computers. Entering an asterisk tells the computer that every file stored in the current ram partition is to be transferred through the RS232 link.

Experiment

From there the two computers take over. If you have a Z-88 with an expanded memory full of files the process may take a long time. Brave users can experiment with changing the baud rate from 9,600 to

INFORMATION:
Program: QZ
Supplier? Sector Software
Price: £25.

19,200 but others may feel that the consequent risk to data integrity is not worthwhile.

Sector Software has cleverly implemented a wildcard file designation system in its program so that sending batches of files to the Z-88 from the QL is just as straightforward.

Some files might be transfer-

red to be used by QL software in which case QZ can be instructed automatically to translate all occurrences of the Z-88 CHR\$(13) linefeed into the QL CHR\$(10) code.

Reluctance

Less cleverly, Batty has promised his customers an automatic translation between *Archive* and *PipeDream* formats but has yet to deliver it. He blames Psion reluctance to reveal the Archive data storage method – surely this is no great commercial secret. When he has this last facility completed,

existing customers can request a free upgrade.

QZ is an essential piece of software for those who own both a QL and a Z-88. It is a well-constructed, sturdy piece of code which I would judge slightly over-priced were it not for the promised Archive translation facility and the knowledge that Batty is always available to give generous support to users who telephone Sector Software. Very few companies will upgrade a program to correct weaknesses without charge and Sector Software is to be commended for its policy.

SOFTWARE FILE

QL World Index

The first regular, independent issue of *QL User*, the predecessor to *Sinclair QL World*, was released in August, 1984. There were 16 issues before it was combined with *Sinclair QL World* and there have been more than 40 issues of magazine since then. This means that there are more than 1,000 articles relating to the QL, its programs, its hardware and its languages gracing the shelves of most ardent QL followers. Many of the features are just as relevant today as when they were hot off the press but, lost among thousands of pages, they might as well have not been written. The QL magazine collector needs an index.

Articles

Some time ago Sector Software approached by Ron Massey to see if there was any commercial interest in a database of QL-related articles which he had developed primarily for his own use. Sector saw the value of the data but insisted on writing its own high-speed database program com-

plied with Digital Precision Turbo to replace Massey's SuperBasic routines. The result is *QL World Index*, a rapid way to find any article printed in *QL World* or *QL User*.

The program just fits on an unexpanded QL, thanks to a variety of data compression techniques. In its fully-expanded form the database would exceed 160K. The user interface with the program is so simple that a manual is considered unnecessary: type-in the word or words to be searched for and the program displays almost instantly details relating to that topic, provided that it was printed prior to May 1988.

As you would expect from Sector Software, the executable program is fully error-trapped and compatible with all mainstream programs. The output shows the name of the magazine, the month of issue, the title of the article, a brief note on its content and the name of the author in a neat tabulation.

Any part of the output can be used as a search key. For example, entering "Aug 1986" will retrieve every article printed in that month's issues;

"Ken MacMahon" will retrieve all articles penned by the former editor and "Psion" will display every article relating to the Psion products. If you are considering buying a new printer for your QL, entering "printer" as a search string will prevent you spending a considerable time flicking through back copies of the magazine and may even save you from buying the wrong printer.

The number of occurrences of a particular theme can be obtained by typing in the word

World and *QL User*. I was surprised by the variety of features the magazine has carried since its launch and by the number of times a vague search string produced the title of the exact article for which previously I had searched in vain.

Lesson

It is not entirely the fault of the program that I was forcibly reminded that swapping discs when data files are open is a dangerous thing to do. Archive users apparently learn this lesson all the time. A warning message in the program display would have saved me much anguish. Better still, the program should load all the data into RAM, provided the space exists.

The rapid display of information is a tribute both to Sector Software and to the power of Turbo. At £6 the program represents very good value, although it is perhaps time for an update to the database to include the most recent issues of your favourite magazine. Do not write to us – write to Sector Software instead.

INFORMATION:
Program: QL World Index
Supplier: Sector Software
Price: £6

"count" followed by the search string. This reveals that there are 25 entries for the April, 1989 issues of *QL World* and that the database holds details of 23 articles written by me.

The value of this utility depends on how much reference use you make of *QL*

operation of Quill. I used *QL Switch* to multi-task on my QL, however, and it was discovered that using version 2.35 of the Psion four did not work when QL Switch was configured for 2.35. This was solved, after consulting Athene, by setting up the switch routine as if Quill 2.35 was version 2.3. If it was set up as 2.35, Quill snatched all the memory and other tasks would not

James McGreehin,
a minister by
trade,
finds that the Lord
needs the QL, too.

activate through lack of memory. Version 2.35 of the other Psion programs works satisfactorily with QL Switch.

Some people have written about key bounce/repeat when typing with the QL. Suggestions have been made about entering code to cope with the problem. Having *ICE*, I alter the Repeat Rate setting in the Custom mode which prevents bounce/repeat, though it slows the response of the cursor a little. Still, you cannot have everything.

When using different daisywheels certain letters and symbols are not given the same code on each wheel and the large printwheel I have has no "I". So in the translates for Quill I set up codes to print "." then backspace, and then print "" which works satisfactorily to produce "I".

The DX-100 has a red printing facility utilising what would be the correction ribbon spool on a typewriter. I have set up the printer__dat so that Highscript prints RED. This means, though, that I have to use different printer drivers for the different daisywheels. Such a problem has been dealt with recently in the magazine but since I had to solve it earlier and did not possess *Editor* I set up a procedure whereby the QL requests selection of printer driver before loading Quill and copies and relevant named driver to RAM1 as printer__dat.

Changes to the driver, once Quill is running, are made using the Key Define facility in the Glossary with TurboQuill+. Having both TurboQuill+ and QL Switch the possibilities for hotkey operation are vast, with TurboQuill+ allowing hotkeys to be set up within Quill. Thus, <CTRL A> changes to the ASCII daisywheel codes:

<CTRL G> changes to the large type
daisy wheel codes:

Spelling checking is done with Qspell, which although it will not multi-task works well and if I save my checking to the end of a work session there is no real problem in re-setting the QL to run Qspell and then running Quill for correction.

One recent suggestion I found helpful was by Dennis Briggs about exporting and importing ordered `__dbf` files to shorten them. It amazed me how big my

R	G	N	F	N	R	R	L	P	E	G	E	E	L	E	K	JEREMIAH
R	G	N	F	N	R	Q	E	J	D	E	G	E	E	L	E	REVELATION
U	R	T	A	I	M	U	B	E	L	E	E	E	E	E	E	ESSEI
U	R	T	A	I	M	U	B	E	L	E	E	E	E	E	E	ZECHARIAH
U	R	T	A	I	M	U	B	E	L	E	E	E	E	E	E	DANIEL
C	T	N	B	X	D	I	R	L	E	E	E	E	E	E	E	JOHN
H	M	L	A	C	H	I	Z	A	S	E	J	E	S	X	F	MATTHEW
H	M	L	A	C	H	I	Z	A	S	E	J	E	S	X	F	EPHESIANS
H	J	A	R	J	R	J	S	J	D	R	A	J	E	H	P	GRATIANS
H	J	A	R	J	R	J	S	J	D	R	A	J	E	H	P	GENESIS
F	R	C	C	M	T	R	L	E	J	E	X	O	S	E	N	EXODUS
L	R	X	A	S	H	L	X	A	S	P	F	C	E	H	E	MALACHI
L	R	X	A	S	H	L	X	A	S	P	F	C	E	H	E	PHILEMON
L	C	N	Y	P	J	R	H	B	E	L	E	E	E	E	E	JOHN
L	C	N	Y	P	J	R	H	B	E	L	E	E	E	E	E	JAMES
L	C	N	Y	P	J	R	H	B	E	L	E	E	E	E	E	RUTH
L	C	N	Y	P	J	R	H	B	E	L	E	E	E	E	E	HEZEKIAH
C	K	N	M	H	E	Z	E	R	A	N	E	E	E	E	E	NUMBERS
J	E	R	E	M	I	N	H	E	Z	E	R	A	N	E	E	LEVITICUS
C	X	G	Y	J	E	B	E	Z	E	K	I	E	L	L	X	ESTHER



church_dbf file had grown, since each time I use it to create the list of who has been visited it selects and orders the file.

The monitor I use was bought later in 1987 because I found that working from a TV set was becoming impossible. I bought a Philips Orange Computer Monitor 80 from Laskeys in Newcastle and it has given no trouble. The facility incorpor-

ated in QL Switch to dim the screen after a few minutes or so no doubt will preserve the screen coating.

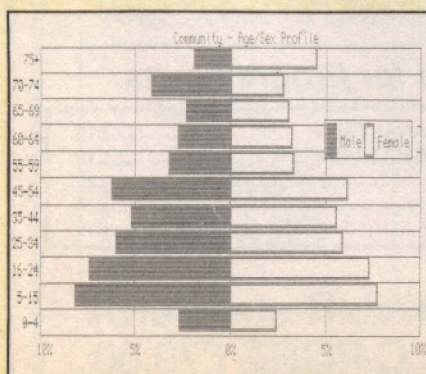
When the article appeared in *QL World* regarding budget disc drives I ordered a drive from D.S. Enterprises. I ordered a power supply from Viglen. With a box and connectors and ribbon cable and the Sandy I/F I had a 5.25in. drive up and running. It has worked well, though I discovered that cheap discs are not always the bargain they first appear to be, with a few failing by slipping in the drive. I find that Tandy and 3M discs work much better.

Recently I bought a 720K, 5.25in. drive from Matmos. The drive works really well and has enabled me to proceed with the latest project, a church magazine. For it I bought the Digital Precision *Desk-Top Publisher* [Special Edition] and the Editor. I found that setting-up the GraFIX driver for the Serial 8056 to print from DTP would prove a problem to me, so I telephoned PDQL, which offered to supply the driver for a small charge.

The program, again in *QL World*, to alter the Gprint_prt supplied with Easel so that it would work with the 8056 allowed me to print from Easel, after I sorted out the error in the listing with the help of a friend who does a good deal of computing. It was while working on data for a church assessment that I found that I could save Easel graphs but when I reloaded them there was no data – only the axes, key and text. I puzzled over this for a few days until I discovered that the problem was that I had named the X-axis as '79 '80 '81 and so on. When I removed the <'> the graph saved, complete with data. I do not know why this fault occurs but it is there, in both versions 2.3 and 2.35 of Easel.

Setting the graph colours to red makes printing with the 8056 better as it creates a uniform dot pattern. Other colours cause patterns which can make the graphs look odd. The finished results were then photocopied on to overhead projector slides and were ready for use. Having spent much time at university drawing graphs I really appreciate Easel, even though I do not use it often.

Another very helpful program I have is *Mailmerge deLuxe* by PDQL. When I bought it I found some odd results, such



Wordsearch dump using GraFIX.

```
1240 DEFine PROCedure CHOOSE_FUNC
1250 k=CODE(INKEY$(-1)):SElect ON k
1260   =232:inp
1270   =236:set_up:start:FILL_COPY:pout (1):save_scr:REMark hard
1280   =240:EDIM
1290   =244:pout 0:save1_scr:REMark hard
1300   =248:S_L
1310   =27:STOP
1320   =REMAINDER :GO TO 1290
1330 END SElect :END DEFine
```

1980 pout (1):save_scr:REMark Hard

Add line 2115, modify 2120 to 2190 as follows:-

```
2115 DEFine PROCedure save_scr
2120 REMark DEFine PROCedure hard
2130 PRINTE0;" SAVE SCREEN FOR PRINTING WITH GraFIX ? <Y/N>"
2140 REPeat YNLOOP
2150 I$=INKEY$(-1)
2160 IF I$=="Y" OR I$=="N":EXIT YNLOOP
2170 END REPeat YNLOOP
2180 IF I$=="N":CLSE0:RETurn
2190 CLSE0:dump_scr:END DEFine
```

Add these lines:-

```
2191 DEFine PROCedure save1_scr
2192 PRINTE0;" SAVE ANSWERS SCREEN FOR PRINTING WITH GraFIX ? <Y/N>"
2193 REPeat YNLOOP
2194 I$=INKEY$(-1)
2195 IF I$=="Y" OR I$=="N":EXIT YNLOOP
2196 END REPeat YNLOOP
2197 IF I$=="N":CLSE0:RETurn
2198 CLSE0:dump1_scr:END DEFine
```

```
1780 REMark DEFine PROCedure dump
1790 REMark CLSE0:CALL where:END DEFine
1800 REMark -----
```

Add these two procedures:

```
3500 DEFine PROCedure dump_scr
3510 CLSE0
3520 DELETE f1p2_Search_scr:DELETE ram2_Search_scr
3530 SBYTES f1p2_Search_scr, 131072, 32768
3540 SBYTES ram2_Search_scr, 131072, 32768
3550 END DEFine dump_scr
3560 REMark -----
3570 DEFine PROCedure dump1_scr
3580 CLSE0
3590 DELETE f1p2_Answers_scr:DELETE ram2_Answers_scr
3600 SBYTES f1p2_Answers_scr, 131072, 32768
3610 SBYTES ram2_Answers_scr, 131072, 32768
3620 END DEFine dump1_scr
```

Modifications to Wordsearch listing from *QL World*, November 1988.

as it printed part of the letter then stopped. Consultation with PDQL by telephone quickly solved the problem. I had set the DIP switches on the DX-100 to give automatic line feed, so line feed was omitted from the Quill printer_dat.

This meant that mailmerge was presented with one long line of letters and spaces. When it reached the limit, which I presume to be about 256, it stopped. Producing the _lis files from Quill with no printer_dat in RAM1 solved the problem and Mailmerge certainly works well. Its facility for single-sheet work, whereby it stops at the end of each page, is invaluable to me since I work with cut paper. If I could pick up a cut sheet feeder

for the DX-100 cheaply my final problem would be solved.

A few years ago I bought *Inkwell deLuxe*. It proved a great help in producing large print hymn and song sheets for those in the church who have sight problems, enabling them to take a greater part in worship. Obviously copywright permission has to be sought before reproducing such material. The church has a licence from the Church Music

Association.

With regard to licences, being a Baptist minister I found that I had to buy my own licence from the Data Protection Registrar, the Baptist Union being unable to register for all Baptist churches. I made the mistake of taking it out in the church's name. This meant that when I moved during the period of the licence I could not take it with me and had to register again. I have done so in my own name now.

All in all, I find the QL an excellent help in my work. In 1986 I knew next to nothing about computers. I certainly did not know how to operate a computer system when I bought the QL but now I would not be without it.

BOMB PROOF

I Dennis Briggs produces the results of his latest researches into crashproofing the QL – for experienced solderers only.

The Sinclair QL computer has been around for some four years during which time it was subjected to at least three major changes before production ceased. After the backlog of unsold machines was disposed of, many owners decided that there was some merit in the old black box despite the profusion of PC clones, along with the incessant publicity from that direction.

Two fairly common problems have arisen with the QL which, despite many owners never encountering them, have plagued some either into parting with large amounts of cash or even the machine to avoid the frustration. The main problem is the use of Micro-drives which can be solved by changing to discs without breaking the bank. The second problem is one of reliability. Let me add that the QL is just as reliable as many computers costing three-figure sums but for serious use it would be pleasant if it could be made 'bomb proof'.

In the past, many theories have been advanced about how to do this with add-on gadgets costing from £20 to £70. While the theorists were theorising, a few workers have

investigated the cause and found the answer.

A small recap in chronological order will provide some of the background. There was very little in the way of quality control in the early days of production but I think that with the passage of time the rogue machines either went back for modification or have been scrapped. The early Psion software, coupled with a dodgy duplication method, also played a major role in frustrating users and again, like the production, quality control was corrected fairly quickly.

One authority stated on several occasions that while the main board was designed to run on 5V, the ULA chips were designed to run on 6V. I have no reason to doubt that but the implementation of corrective action is out of the question.

Explained

The power supply designer was consulted and explained why the standard unit was not up to the job; he also explained that a suitable one of the many hundreds he has designed would be much better. This one has been altered slightly; with large sales proving that it

is effective in many situations. A similar situation arises in regard to spike suppressors or power conditioners, which again help to a certain extent, as does the replacement of the 5V regulator with a larger rated one inside the machine.

When expansion boards became affordable the problems increased because of several factors, mainly in relation to a programmed chip on the expansion having some conflict with the 8301 ULA. Fitting one of the other four different 8301 chips usually cured it to a large extent.

Remedies

Two very knowledgeable QL enthusiasts to whom I am indebted have looked at the problem of random crashes and have produced the following remedies. Let me stress that the work involved should be attempted only by those with sufficient knowledge and soldering skills.

1. Replace the 7805 with the 2amp 78S05.
2. Check that the voltage at the 68008 is 4.9V.
3. Put a 0.1mfd capacitor between the centre and right-hand pin of the 78S05.

4. Replace the long jumper wire from the regulator to the thick track just below the 9-way membrane connector with solid cored insulated wire with two or three ferrite beads on it.
5. Connect a 0.01mfd capacitor between pin 6 and pin 15 of the 8301, then a 0.001mfd from pin 6 of the 8301 to pin 15 and pin 35 of the 68008.
6. Solder a 0.01mfd capacitor in parallel with a 10mfd 10V tantalum capacitor and connect to pin 20 and pin 40 with + to pin 40 of the 8309 co-processor.

Options

- A. Change IC19, IC20, IC21 and IC27 for their HCT equivalents.
- B. Connect a 68pf capacitor between each data line and ground.
- C. Replace the 0.01mfd capacitors between each RAM chip with 0.33mfd if the RAM chips are the slow 200ns types.
- D. Fit a 4.7mfd tantalum capacitor across the supply pins of every fourth RAM chip in the QL.

With a small outlay and some careful work you will have a QL which is completely reliable and crash-proof.

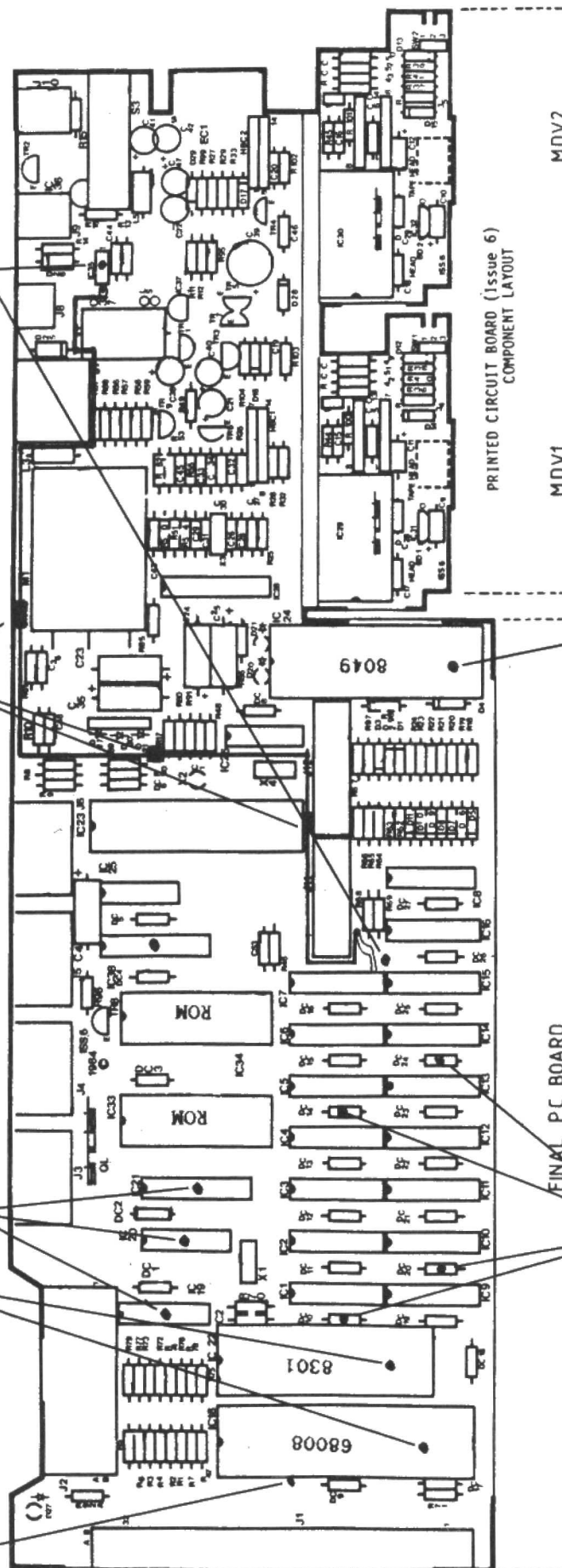
Solder a 0.22uF from pin 13 to pin 15 of the 68008.

Connect the CPU ground to the 8301 ground with a 1000pF capacitor.

Replace ICs 19, 20 and 21 with HCT types or solder a 200pF capacitor from pin 8 to pin 14.

Ferrite beads

Thick solid cored wire with two or three ferrite beads from 5V regulated output to the RAM supply track from here to here.



FINAL PC BOARD

MDV1

MDV2

Fit a 4.7uF tantalum to decouple every fourth memory chip.

Solder a 0.1uF capacitor from pin 28 to pin 40 of the 8049.

This article continues our exploration of QL and Thor system ROMs, the program chips which allow the computer to recognise and act on commands. In recent years I have documented more than 100 bugs in QL ROMs, from version AH to MG. This month I look at improvements and corrections in new ROMs from QView and Thor International. It is not all good news. I hope to excite veteran bug-hunters with tales of a few newly-detected QL faults.

Minerva is the name of the Roman goddess of fortune and a QL ROM upgrade from QView. It is a plug-in replacement for the two Sinclair ROM chips located on the QL circuit board behind the cartridge socket. Once Minerva is fitted in your QL it cures bugs, adds new features and speeds the system.

Minerva still contains some original Sinclair code, so QView has had to be careful to avoid infringing copyright. It supplies the upgrade as a back-up of your normal QL ROM, with improvements. You must prove that you own a QL by sending an image of your original ROM, easily created with the command `SBYTES filename,0,49152`.

When QView receives your disc or cartridge, along with £30, it copies documentation on to the medium and supplies a tiny circuit board holding two chips. The biggest chip is a 64K EPROM which holds the new operating system. Below is a logic chip which makes the EPROM seem like the Sinclair 48K ROM.

The remaining 16K of EPROM space is disabled, so you can use the external cartridge port as normal. You can extend the chip contents with an EPROM programmer and enable all 64K by cutting a link but if you do so you must not use the ROM socket on the back of the QL.

Fit with care

Minerva is easy to fit, so long as you are careful. Disconnect the power, then open the QL case by undoing eight cross-point screws in the bottom of the box. The two screws holding the Microdrives should be left alone. Next you must prise both Sinclair ROM chips out of their sockets with a screwdriver or chip extraction tool. Be careful not to prise up the socket as well, or you will need a new circuit board.

Minerva plugs into either of the empty sockets, so long as your internal RAM or keyboard interface does not get in the way. It is easy to plug in the unit but hard work getting it out if you need to put back your old ROMs for some reason. Once the keyboard is screwed in place you can connect power and you are ready to go.

The documentation consists of text files and short Basic and machine code programs. You can read the text and code by copying the files to the screen, importing into Quill or using a text editor. You get about 15 A4 pages, with an introduction, fitting notes and sections covering con-

MINERVA THE ROM

cepts, Basic, assembler, and known incompatibilities, which are remarkably few considering the number of improvements.

I shall discuss the features under three headings – innovations, speed-ups and bug-fixes. The innovations split logically into two groups – improvements to Basic and device enhancements.

Minerva SuperBasic interpreter has had a major overhaul. At last you can use integer and string values in `SELECT` statements. Sub-ranges and multiple instances are allowed, in single or multi-line commands. You can already compile these with Turbo or the latest version of *Q-Liberator* but now you can test programs interactively which use them.

Minerva lets you write integer and even string `FOR` loops, with all the flexibility of floating point loops but a limit of four characters on string instances. Integer loops save time, especially in array indexing.

String `FOR` loops are a little odd as they treat characters as signed byte values but they are useful at times, such as when printing the Minerva extended character set – figure one – Integer `FOR` loops can be compiled with Turbo and Q-Lib but neither supports string `FOR`.

3D and 2D calculations are easier because of improvements in ATAN and ABS. ATAN accepts one or two parameters. If you give it X and Y separately it converts rectangular co-ordinates into polar form, avoiding overflows and quadrant errors. ABS can take multiple para-

“Minerva is the name of the Roman goddess of fortune, and a QL ROM upgrade from QView.”

meters, returning the square root of the sum of their squares. Pythagoras would have approved.

Other changes improve `DATE`, `FILL`, `MODE`, `PAUSE`, `RESPR`, `SCALE`, `SDATE` and `VER$` without harming programs which use them in the normal way. There are some new vectors for machine-code programmers but they are useless if your code must work with Sinclair ROMs as well.

The keyboard editor, `10.EDLIN`, has been improved; the Space key and Enter

Simon Goodwin looks again at the bugs and features of QL-compatible systems.

work even if you press Shift, as in Quill. Esc aborts the editor and Alt Left and Right move the cursor to the start and end of the text.

Shift Tab and Tab move the cursor back and forth in eight character steps and you can delete to the end of the line or to the start of the window in one step. `10.EDLIN` is used by `EDIT`, `INPUT` and functions like `DIY Toolkit EDLINE$`.

You can ‘compose’ accented characters on the fly, with no need to look them up in the QL *Concepts* guide. Type Ctrl-Enter, then the required letter, then the type of accent – ‘/’ for an acute accent, ‘’ for grave, colon for amulet, ‘ã’ for circumflex, and so on.

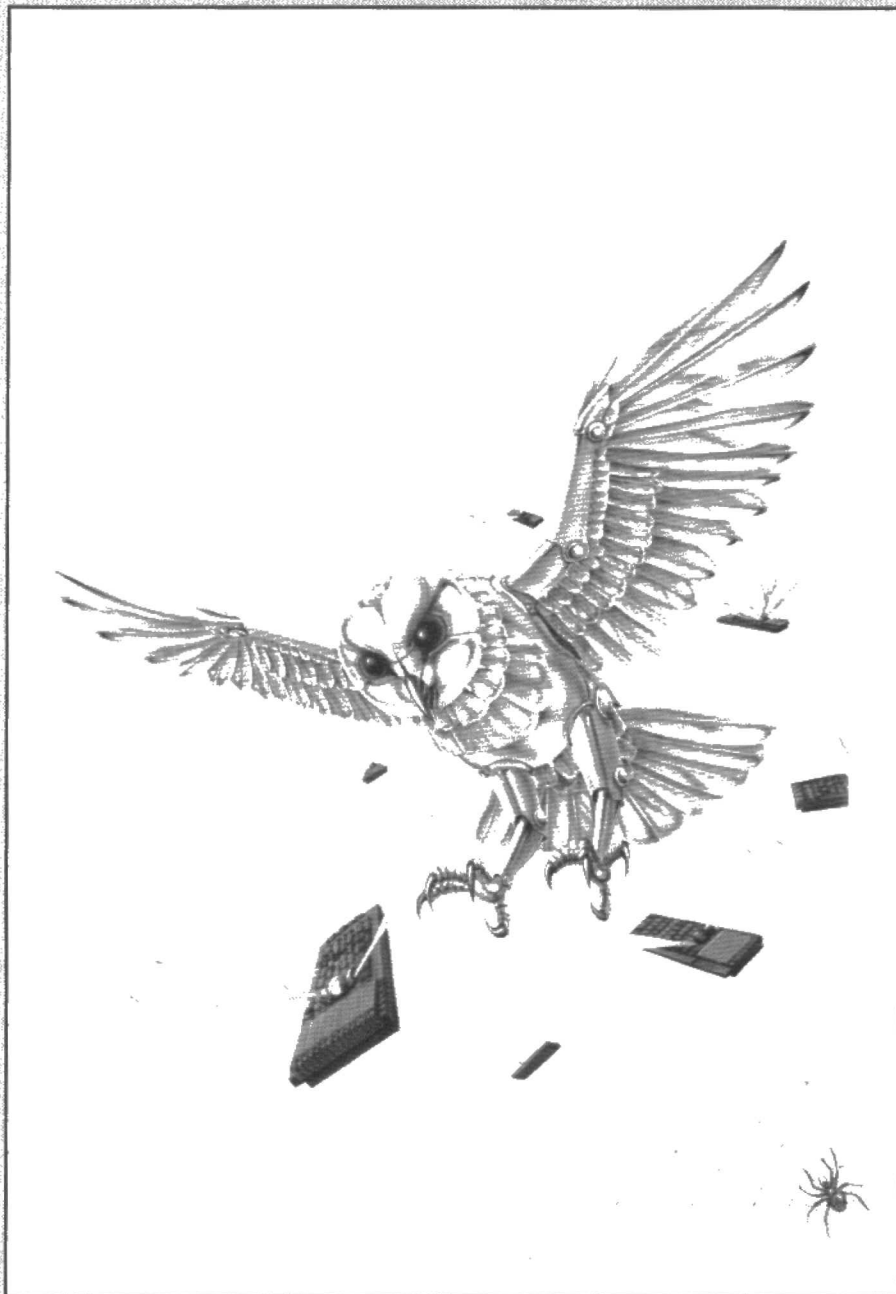
You can type the letter and accent in either order or use the old keystroke. Caps Lock and Shift work as normal. There are 50 possible combinations, including symbols as well as accents. To type an arrow, press Ctrl Enter, then the arrow key.

The Microdrive handler no longer locks up the machine when RAM is scarce. Minerva marks files with the data when they are created or updated. Unfortunately disc ROMs and *SuperToolkit 2* override this code.

The QL hardware is designed to allow two areas of screen memory, either of which can be displayed at any time. They can both be in the same mode or one can be in `MODE 4` and the other in `MODE 8`. Switching between the two screens is instant; in essence, all it takes is a single `POKE` to the display master chip.

Sinclair never implemented the second screen and it is not possible to use it with Sinclair versions of Qdos because vital system information is stored in part of the memory allocated to screen 2. Qdos would crash as soon as you cleared the second screen.

QView has fixed Minerva so that both screens can be used if you press F3 or F4 after a re-set. A new `MODE` command lets you look at either screen, in `MODE 8` or



MODE 4, and direct new console channels to either screen. Control Tab swaps the display from one screen to the other; alternatively the screens swap automatically as you switch between cursors.

This works satisfactorily in SuperBasic, once you are used to the arcane complexity of the new MODE command. Unfortunately very few compiled programs can cope with the second screen and the relocated system information.

Stronger string

Some programs assume the address of the system tables and corrupt the second screen display. Psion Quill almost works but becomes confused when you try to redraw a screen. It puts the text in the expected place but clears the other screen and draws empty boxes there.

Turbo and Supercharged tasks fail because the system variables are not in the usual place, so they cannot find Basic

keyword addresses. Similar problems affect the Q-Liberator library.

The second screen is a good idea but it is incompatible with most QL software. This should change with time as programs are re-compiled. Revised Turbo and

"Minerva tests RAM at about twice Sinclair speed, so the QL starts faster . . . The memory test can be skipped once you have started the machine."

Q-Lib code is being developed but it may be a little time before popular applications are compatible with the new mode. In the meantime you can use two screens while working in Basic and, anyway, Minerva

has plenty of features which are compatible with existing software.

Minerva tests RAM at about twice Sinclair speed, so the QL starts faster as soon as you turn it on. The memory test can be skipped once you have started the machine. You can re-start the system quickly with a CALL command or by pressing Shift Ctrl Alt Tab.

Early QL software is upset by expanded systems, so Minerva ignores RAM expansion if you press Shift at the start. Similarly, Ctrl prevents it initialising ROM devices and toolkits. If you do not press a function key Minerva assumes F2 after 30 seconds.

Minerva has improved graphics which run lightning-fast with no RAM overhead. The speed-up also affects ellipses which are not accelerated by add-on software. Lines, points and turtle graphics are drawn much faster and Minerva corrects bugs in FILL, arc and ellipse handling. For a laugh, try this on a QL or Thor:

```
FOR r=70 to 99: ELLIPSE 50,50,r,6,1
```

Character output is not accelerated but Minerva is compatible with *Speedscreen* and allows eight pixel-wide characters in the maximum width, unlike Sinclair ROMs.

SuperBasic programmers will notice speed-ups in the interpreter. Loading can be 25 percent faster. Contrary to Sinclair claims, SuperBasic runs more slowly as programs grow in size. Loops and definitions are found by scanning the program, line by line. Minerva does not eliminate the delay but it scans about twice as fast as Sinclair code.

Almost works

String handling is faster and more reliable. The interpreter no longer moves strings needlessly. Errors during concatenation, like this:

```
PRINT "x" & (1/0)
```

no longer crash Basic, as they do on the QL and Thor.

Minerva supports true integer arithmetic and fixes the bugs associated with the value -32768. Integer handling is now slightly faster than floating point maths, rather than the other way round as on earlier ROMs. The speed-up factor is about 40 percent at best.

Floating point maths is accelerated but it is difficult to tell the difference unless you use a stopwatch. Minerva speeds multiplication and division in Turbo and Supercharged tasks but does not accelerate current versions of Q-Liberator as they do not call the ROM in such cases. Transcendental functions are faster, whether interpreted or compiled.

All operating system calls have been streamlined. This speeds programs which

The speed-ups in Minerva are many but often subtle. Almost every program shows some speed improvement but arguably the most important time-savers are the corrections which prevent the machine crashing at odd moments.

Calls to deleted procedures no longer crash Basic. You can safely use more than nine parameters or locals in a definition. EDIT works after a 'not complete' report. Memory is less likely to

```

WHEN x>511
  PRINT #0;"Horizontal overflow."
  x=511
END WHEN

```

Most QL MERGE bugs are fixed by Minerva but you get the report 'not

Another program, written in SuperBasic with Toolkit 2 commands, performs rudi-

```
FOR c%=CHR$(-128) TO CHR$(127):print !c%!
```

0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\] ^ _ ` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~

The MT.ALCHP system call is used to accept negative parameters when allocating memory on the system heap. Thus the Toolkit commands ALCHP and ALLOCATION could crash the machine if given negative parameters. Like Minerva, last month's DIY Toolkit function RESERVE is always safe, as it rejects a negative parameter.

QDOS needs a buffer of more than 1K to keep track when you use FILL in a window. If you close the channel before executing FILL 0 the memory is lost for ever, unless you are using Minerva. Sadly the bug is restored if you have QPTR loaded.

WHEN ERROR and WHEN Variable had terrible bugs under the Sinclair interpreter but work well under Minerva. WHEN ERROR can intercept any report, whereas error-trapping during calculations used to crash the QL and Thor.

implemented yet' if you try to MERGE from inside a procedure. Sadly, SAVE still fails to report an error if it fills a device or you press CTRL-SPACE before it has finished.

Minerva fixes the bug which locks the QL if you close a network channel without writing to it but unfortunately SuperToolkit 2 over-rides the fix and re-introduces the bug. Network broadcast now works, so long as all the machines have Minerva fitted.

A few benign bugs remain; SCROLL 42 still allows random access and PAN still lets you turn cursors on or off.

The first version of Minerva returned VER\$ as "JSL" - standing for "Jonathan", "Stuart" and "Lawrence", the team at QView. Most of the work was done by Lawrence Reeves. The Qdos version returned is 1.61. It had a few new bugs of its own but they have been fixed in 1.62 and the latest 1.63. VER\$ "JSL1"

Several programs are provided to circumvent bugs in existing software which become obvious under Minerva. A 'bodge' program fixes old Q-Liberated tasks to run under Minerva and the MG ROM. Another program fixes bugs in Hotkey versions 2.06 and earlier.

mentary checks to see if a program is likely to work in the two-screen mode.

The first version of Minerva had a few 'new' bugs but version 1.63 is reliable enough to be fitted permanently in my work QL. So long as you avoid the two-screen mode it enhances almost all commercial QL software. This is a remarkable achievement and Reeves is to be congratulated for fixing so much without introducing new problems. Minerva is a big step up from Sinclair Qdos and good value at £30 for 48K of EPROM.

Meanwhile, in Denmark, Thor International has revised *Argos*, the QL-compatible operating system of the Thor XVI. The first Thor 8 machines ran version 4 of the system, based on Sinclair JS ROM and QL parts. Version 5 was used in the Thor 20 and now we have Argos version 6 on the XVI.

The first 'solid' version of Argos was 6.30, VER\$ "PT", named after David Oliver's wife Penny Tzatzaris. Despite its 'virtual clean room' derivation it had most of the bugs of the QL MG ROM. Minor tweaks followed and a much-improved hard disc device driver was introduced for version 6.37.

Version 6.40 was a disaster, introducing scores of new bugs, the worst of which were banished in 6.41. Development is notionally 'frozen' at 6.41 but in practice I have seen two versions with this number, PQ and CS, with slight differences.

Sadly, you cannot run Minerva on the Thor but the Danes have fixed one bug which afflicts all QLs:

```
DIM X(4) : INPUT X
```

It gives a 'not implemented' error on Minerva. QL ROMs just read one value and discard it but the Thor patiently accepts five values, for X(0) to X(4). Disappointingly, DIM Y(4) : Y=X gives 'operation not yet implemented'.

There is still plenty of scope for ROM developments. Watch this space and please write with full details if you find new bugs.

The Archive Screen Driver

In a further investigation into the Archive screen driver and its codes, Mark Coulthard relates the Archive commands to IBC PC-Four commands and produces some procedures to implement them.

The power and potential of the Archive database program is well-known to those who have taken the time and effort to explore the database language and apply it. Having used several card index-type database management systems on IBM clones, Archive is far more impressive. Its ability to have several files open at once allows relational databases to be implemented and used according to your own program. It is the programming language which allows flexibility to control data and also drive pen plotters for high-quality output.

The QL screen displays can lack the professional appearance often found on quality programs on IBMs. This is due mainly to the use of the IBM graphics box

characters available on IBM PCs. These box characters can be used with good effect within the Psion Xchange suite or Psion PC-Four running on a PC.

Development

The development of a database for the recording of borehole data, which required seven files to be open at once, led to an examination of the screen driver used in PC-Four. The development of the program was carried-out at work on an IBM and at home on a QL with programs and files being exchanged by linking QL to IBM via the RS232 port.

The use of graphics characters and window handling on both machines was therefore imperative. On testing the QL Archive version 2.3 it was a pleasant

surprise to find that the same commands can be used on QL Archive as on IBM versions of PC-Four.

The QL Archive uses its own font which has graphics characters in ASCII code starting at 224—table 1. These characters are accessible both from the keyboard or by using the PRINT CHR() from the Archive language. Any attempt to use these from Basic using CHR\$() will not work.

Dashed

The possibility of using these characters in screens loaded from Archive was dashed when it was realised that the character (r), the top left box corner, was generated by the F1 key which unfortunately is also the help key. Thus a complete

TABLE 1

Keyboard	Character	ASCII
Caps Lock		224
Alt Caps Lock		225
Ctrl Caps Lock		226
Alt Ctrl Caps		227
Shift Caps		228
Shift Alt Caps		229
Shift Ctrl Caps		230
Shift Ctrl Alt Caps		231
Ctrl F1		232
Ctrl Shift F1		233
F1		234

TABLE 2

Value	Colour (Mode 4)	Colour (Mode 8)
0	black	black
1	blue	black
2	red	red
3	magenta	red
4	green	green
5	cyan	green
6	yellow	white
7	white	white

box cannot be drawn when editing screens from within Archive. The results, however, are acceptable when using the PRINT CHR() command.

The presence of the graphics characters is just the tip of the iceberg, since there are 24 other screen driver codes. The most useful one enables windows to be defined from within the Archive language, thus bringing screen displays on a par with SuperBasic. One disadvantage of the graphics shapes on the QL is that they do not fit the character square fully and thus solid lines cannot be obtained.

The Archive screen driver allows screen displays to be controlled by passing control codes to the screen driver from a program using the PRINT CHR() command. All control codes are within the range 0-31, i.e., PRINT CHR(20). Some

ter to be repeated; the second is the number of repetitions, i.e., PRINT CHR(4)+CHR(231)+CHR(10) will produce the top edge to a box.

Cursor control codes

A large number of control codes are used to manipulate the cursor from within the print statement.

- a) Underline : Driver Code 5: No parameters – toggle switches between underline on and off.
- b) Cursor Right : Driver Code 6: No parameters – moves the cursor right by one character space.
- c) Cursor Left : Driver Code 8: No parameters – moves the cursor left by one character space.
- d) Tab : Driver Code 9 : One parameter – similar in effect to the tab statement from Basic. It moves the cursor to the column specified in the parameter printing spaces in the process; i.e., PRINT CHR(9)+CHR(15) will move the cursor to the fifteenth column in the present window.
- e) Line Feed : Driver Code 10 : No parameter – moves the cursor down one line.
- f) Cursor Up : Driver Code 11 : No parameter – moves the cursor up one line.
- g) Form Feed : Driver Code 12 : No parameters – this clears the screen and homes the cursor.

One should remember that when this is executed from the print statement, the print statement will issue a line feed automatically and therefore a semi-colon should be used to prevent this.

- h) Carriage Return : Driver Code 13 : No parameter – moves the cursor to the left-hand margin of the window on the same line.
- i) Cursor On : Driver Code 14 : No parameter – switches on the cursor.
- j) Cursor Off : Driver Code 15 : No parameter – switches off the cursor display.
- k) Home Cursor : Driver Code 30: No

parameter – moves the cursor to the top left corner of the window.

- l) Position Cursor : Driver Code 31: Two parameters; the two parameters are used as the x-co-ordinate; the y-co-ordinate and its action is similar to the at command.
- m) Delete Character : Driver Code 19: No parameter – moves the cursor one character space to the left and prints a space, thus deleting to the left of the cursor.

Window commands

The ability to define a window from Archive is probably the most useful.

- a) Window definition : Driver Code 20: four parameters. This command defines a window in character co-ordinates from the top left-hand corner of the screen. The window is obtained by defining the origin of its top left corner using the first two parameters, and its size with the next two – tables 3.

```
PRINT CHR(20)+CHR(15)+CHR(5)+CHR(35)+CHR(15)
```

This gives a window starting at position 15,5 and having the size of 20 columns wide and 10 lines deep. Additional codes are also used to alter the control of the window scrolling.

- a) Scroll Screen Up : Driver Code 21: One parameter – scrolls the screen up by the number of lines specified by the parameter.
- b) Scroll Screen Down : Driver Code 22: One parameter – scrolls the screen down by the number of lines specified by the parameter.
- c) Set Boundary Type : Driver Code 25: One parameter – the parameter sets the cursor behaviour at the edge of the window with each bit of the parameter controlling one specific aspect – table 4.
- d) Swap ink and paper colours : Driver Code 26: No parameter – exchanges the ink and paper colour.
- e) Carriage return and line feed : Driver Code 28 : No parameter.

TABLE 3

Parameter	Definition
1	x-coord Top left
2	y-coord Top left
3	x-coord Bottom right
4	y-coord Bottom right

of the control codes require additional parameters again passed using chr(). Failure to comply with the correct codes generates an error number 39 I/O failure.

Paper/Text Control Codes

- a) **Text Colour** : Driver Code 1:1 parameter.
The parameter normally takes the value 0-7 and thus bits 0-2 of the byte can be set. The value defines the colour as shown in table 2. If bit 7 of the parameter is also set – i.e., numbers 128-135 – the current text colour is saved and the following text is printed in the temporary colour, i.e.;

```
PRINT CHR(1)+CHR(2)+"AAA"
PRINT CHR(1)+CHR(132)+"BBB"
PRINT "CCC"
```

These lines will print three red As followed by three green Bs and three red Cs.

- b) **Paper Colour**: Driver Code 2:1 parameter.
This code sets the paper colour as per table 2. If bit 7 of the parameter is set the current paper colour is saved and the specified paper colour is set temporarily.

- c) **Repeat Characters**: Driver Code 4 : 2 parameters.
Similar to the SuperBasic command REPT. The first parameter is the charac-

TABLE 4

Bit	Action when set	Action when clear
0	auto scroll up	no scroll up
1	auto scroll down	no scroll down
2	cursor wrap right	no wrap
3	cursor wrap left	no wrap
4	wrap on same line	wrap cursor onto next line
5	wrap on same line	wrap cursor onto next line

f) Escape Sequences : Driver Code 27 : One parameter – provides a range of facilities most of which are covered by the other codes.

in procedures. The procedures shown in listing one demonstrate the use of some of the codes. The three procedures TEST, WINDOW_B and WINDOW_NORM

a window according to the parameters passed to it. The parameters are almost the same as for Driver Code 20 except that the window size, in columns and lines are requested i.e., the call WINDOW_B;15,5,20,10,2 produces a window starting at column 15, line 5 and is 20 columns wide and 10 lines deep and colour 2. The window is produced with a shadow effect and a border using the graphics characters. The window inside the border thus scrolls without affecting the border.

The TEST procedure calls WINDOW_B and writes text to the window in a variety of ways; note that the procedure can be run in either MODE 0,4 or MODE 0,8.

Parameter (ASCII value)	Result
65	Clears from cursor to end of line.
66	Clears from the end of window.
67	Saves the cursor position.
68	Restores previous cursor position.
69	Scrolls up one line from top of window to cursor.
70	Scrolls up one line from cursor to bottom of window.
71	Scrolls down one line from top of window to cursor.
72	Scrolls down one line from cursor to bottom of window.

All these commands can be used in combinations to produce quality screen displays and handling and are used best

are activated by running TEST from the Archive environment.

WINDOW_B : This procedure defines

LISTING 1

```

proc test
  local y
  paper 7 : cls
  window_b;15,5,20,10,2
  print "this is some text" : print : print : print "and some more"
  print chr(30) : rem home cursor
  let y=1
  while y<150
    let y=y+1
  endwhile
  let y=1
  print chr(2)+chr(7)+chr(1)+chr(0) : rem set paper and ink colour
  cls
  while y<=7
    print y ; " HELLO " : print chr(26)+" HELLO "
    print chr(2)+chr(7-y)+chr(1)+chr(0+y);
    rem scan through the paper and ink colours
    let y=y+1
  endwhile
endproc

proc window_b;cst,lst,wcol,wline,col
  local x
  let x=2
  rem define shadow
  print chr(20)+chr(cst+1)+chr(lst+1)+chr(cst+wcol+1)+chr(lst+wline+1)
  paper 0 : cls
  rem define window
  print chr(20)+chr(cst)+chr(lst)+chr(cst+wcol)+chr(lst+wline)
  paper col : ink 0 : cls
  rem print top boundary of box
  print chr(234)+chr(4)+chr(231)+chr(wcol-2)+chr(226)
  while x<=wline-1
    print chr(224)+chr(9)+chr(wcol)+chr(224); : rem print sides of box
    let x=x+1
  endwhile
  print chr(25)+chr(0) : rem set boundary
  rem print bottom edge of box
  print chr(227)+chr(4)+chr(231)+chr(wcol-2)+chr(233);
  rem def print window
  print chr(20)+chr(cst+1)+chr(lst+1)+chr(cst+wcol-1)+chr(lst+wline-1)
  paper 4 : cls
endproc

proc window_norm
  print chr(20)+chr(0)+chr(0)+chr(80)+chr(23)
endproc

```


SOFTWARE FILE

The Blag 2

INFORMATION

Program: The Blag 2.
256K minimum memory.
Supplier: CGH Services,
Cwm Gwen Hall,
Pencader, Dyfed, Cymru
SA39 9HA.
Price: £8, disc or
Microdrive.

This adventure was published originally by GAP Software but when it ceased trading, the author, Tony Woolcock, arranged for CGH Services to publish an updated version, with many of the earlier criticisms rectified.

Adventure

The adventure is not the ordinary run-of-the-mill fight against fiends and monsters in some kind of fantasy world but is instead a fight against bureaucracy and the criminal fraternity. There has been a big robbery at the local bank and although the robbers were chased by one brave citizen who managed to get the number of the get-away car, it is your job to identify the robbers and to apprehend them.

So, armed only with your truncheon, helmet and notebook, you find a memo left by the superintendent welcoming you to your new job and are thrown into the world outside the police station.

There seems to be no way of getting killed in this adventure but mishandling the situation may mean that you ruin your chances of completing it. Handling evidence without taking precautions against fingerprints will render the evi-

The Blag 2 is a battle of wits, finds Rich Mellor.
“... excellent, forming a breakaway from the rather tired format of ordinary adventures...”

dence useless, so you need to proceed carefully. Although there is a mastiff guarding some gates which will not let you move, there is an easy way round any fun-loving dog if you think about it. All the problems in the adventure are very logical and once thought of in terms of what the police would do, present no real difficulty.

There is help in the form of a police computer which holds collaterals' records, criminal and vehicle records, a crime complaint file and a scene-of-crime office. There is also a police dog which can help once you find his name. The dog is easy to control by commands such as 'SEEK' and 'FIND', which make it smell the air for scents and point in a certain direction.

Police

There is also a police car to help you to reach locations faster. This, too, is very easy to control, since 'DRIVE CAR' and 'PARK CAR' are all that are needed; once you are driving the car, you use the normal direction commands. Beyond that very limited help you are on your own and must look for clues in much the same way as a detective would investigate a crime.

To give you a little more help

in solving the crime you can question the characters who are strewn all over the place and ask about different aspects of the case. If you ask about the 'robbery', the eye-witnesses will tell you what they saw. Not all characters you meet like the police, so a little tact is needed to obtain information from them.

Input

A novel feature is the use of three windows to display information. The main window contains a description of the location; another lists all the objects both present and carried by you and the third window is used for you to input commands and for the computer to respond to those commands. Some commands are made redundant, such as LOOK and INVENTORY, since the information is always present. Luckily for many QL users, despite the number of information windows the game fits well on a TV screen, which was one of the major criticisms of the original version.

Certain actions, such as questioning suspects, or using the telephone, call up a fourth pop-up window which overlaps all the other windows and is used for commands specific to that action. When questioning

someone, the questions you wish to ask along with the answers appear in this window. It does not slow the game any – it probably speeds interpretation of commands – but unfortunately must contribute to the use of memory.

WRITE or READ NOTES will show another feature of the game, “Blunder's notebook”, which enables you to make short notes on different aspects of the case and save the notes on Microdrive or disc. It seems as if this notebook can hold a good deal of information, media space permitting, and therefore obviates the need for pencil and paper.

Successful

The game also features a large database of police trivia which is displayed at each location. That unfortunately does not provide hints towards the successful completion of the game but gives it an added dimension. It can be a little annoying and perhaps the ability to turn this feature on and off would be appreciated.

There were plans to produce a *Blag 3* on the QL, which was to contain more crimes to solve and include graphics and other little extras. It seems it will now appear only on the ST, filling around three discs. In this, Woolcock is following the footsteps of many software authors on the QL.

My only criticism of the game is that it contains a few minor spelling mistakes but otherwise it is excellent, forming a breakaway from the rather tired format of ordinary adventures. It is excellent value and I would recommend it to anyone who enjoys a little detection.



SUPER BASIC

This month completes last month's file management project with file and device matching routines.

This month's article contains listings which complete the file management project developed to demonstrate the pattern-matching routine published recently in this series. Last month's article concentrated on the presentation of information, whereas this month's routines are more concerned with the behind-the-scenes manipulation of files and devices.

The structure which binds together the accompanying definitions is simple. The most important routine printed here is listing 16, *Search_Mode*, through which all the remaining routines are reached from the main program.

Three functions are immediately subordinate to *Search_Mode* and three proc-

edures, one with a subordinate function, are reached through the *File_Check* function. This simplicity is repeated in the majority of the individual definitions, which is a tribute to the strengths of a structured programming language used in an effective manner.

Puzzle

Nevertheless, this program set a puzzle which seems to have no straightforward solution. File directories can vary from just two entries on an under-utilised Microdrive to more than 100 on a disc full of short files. When a large directory listing is displayed on the screen the window scrolls to allow more information to appear. When the same information is

directed to the printer, however, the length of a piece of paper remains stubbornly the same.

To control output to the printer the concepts of page length and page breaks have a large influence on the design of the program. The main data arrays are designed to hold one full page of data and the *Search_Mode* procedure contains a FOR...NEXT loop which is repeated once for each line of output. If the loop terminates before a listing is complete the listing is interrupted by a page break.

Users must be given not only the option to print a page when it becomes full; they must also be able to select a print at the end of each directory listing.

It would be wrong for the program to compel users to obtain printouts at the end of each directory. If a user is searching for a dozen related programs saved on three or four discs, for instance, it would be appropriate to place the output on to a single page.

Portions

The essence of the problem is that input is obtained in irregular-sized portions while the output must conform to a fixed maximum page length. This inconsistency must be modelled accurately within the *Search_Mode* procedure. The procedure is unlikely to be of much relevance to any other program and therefore only its general design is likely to be of interest here.

Search_Mode is designed round four concentric loops with not a little of the logic consigned to slave functions. Slave routines are those called by only one routine in a program and which are designed to simplify the operation of their masters. All three functions return binary values, one or zero, depending on whether an option has been selected by the user.

This technique simplifies the main code by subcontracting to the slave routines not only an optional task but also the menu operations associated with it. If the first slave function – listing 17 – had been written as a procedure it would have obtained a directory listing from a file

Listing 16

```
1600 DEFine PROCedure Search_Mode
1602 LOCAL Loop1, Loop2, Loop3, X, Y
1604 Y = 1
1606 REPEAT Loop1
1608   IF Fetch_Dir = 0: EXIT Loop1
1610   REPEAT Loop2
1612     Menu 0
1614     FOR X = Y TO DIMN(File$)
1616       REPEAT Loop3
1618         IF EOF(#T): EXIT X
1620         INPUT#T, a$: BEEP 200, 100
1622         IF Match(Pattern$, a$): EXIT Loop3
1624       END REPEAT Loop3
1626       File$(X) = a$: Media$(X) = Medium$
1628       PRINT Medium$: TO 12; a$
1630       IF Examine
1632         IF File_Check: CLOSE#T: EXIT Loop1
1634       END IF
1636     NEXT X
1638     PRINT"End of Page"
1640   END FOR X
1642   IF Print_Page OR X = DIMN(File$)
1644     Y = 1: Display
1646   ELSE
1648     Y = X
1650   END IF
1652   IF EOF(#T): CLOSE#T: EXIT Loop2
1654 END REPEAT Loop2
1656 END REPEAT Loop1
1658 Display: Menu 1
1660 END DEFine Search_Mode
```


Listing 17

```

1700 DEFine FuNction  Fetch_Dir
1705 LOCal Loop,  a$
1710 Menu 4.1:  IF Bar_Menu (2) <2:  RETurn 0
1715 DELETE      TempFile$
1720 OPEN_NEW#T,  TempFile$
1725 DIR#T,      Dev$
1730 CLOSE#T:    OPEN_IN#T, TempFile$
1735 INPUT#T,    Medium$,  a$
1740 Free = INT(a$/2)
1745 Used =a$("/" INSTR a$ +1 TO " " INSTR a$ -1)
1750 Used = INT (Used /2) - Free
1755 Medium:    Space:  RETurn 1
1760 END DEFine Fetch_Dir

```

Listing 18

```

1800 DEFine FuNction  Print_Page
1805 Menu 4.2
1810 IF Bar_Menu(2) = 2
1815 PRINT#P;  \\ DATE$
1820 PRINT#P;  \  "Files Matching ";
1825 PRINT#P;  Pattern$ \\
1830 FOR X = 1 TO DIMN(File$)
1835 IF File$(X) = "": EXIT X
1840 PRINT#P, TO 8; Media$(X);
1845 PRINT#P, TO 24; File$(X)
1850 END FOR X
1855 PRINT#P, CHR$(12)
1860 RETurn 1
1865 END IF
1870 RETurn 0
1875 END DEFine Print_Page

```

storage medium. As a function, however, it can also include the menu which asks the user if a directory listing is to be obtained. If the user declines the offer the function is aborted and a zero is returned; otherwise the function completes its task and returns a one.

Many stylists argue that the job of a function is to convert arguments into values and that all other forms of processing belong to procedures. Similarly, they claim that it is not the business of a procedure to alter a value passed to it. It may be a good yardstick by which to measure the quality of a piece of code but, as with all rules, there are many valid exceptions. Practical programmers cannot afford to be pedants and the principles of accuracy, brevity, consistency, dependability and efficiency frequently over-ride such dogmatic regulation.

Clear decks

Even after clearing the decks of much of the supporting code the *Search_Mode* procedure remains complicated. In essence, the outer loop cycles once for each directory listing. At the end of each cycle the user is offered the opportunity to print the filenames so far collected. Leaving aside the second loop for a moment, the FOR...NEXT loop cycles once for each correctly-matched filename found in the directory. If the FOR...NEXT loop identifier reaches its maximum value the user is offered the print option. Whether

the user accepts the offer or not, the arrays must be re-initialised before any more filenames can be collected.

The FOR...NEXT loop is exited prematurely at line 1618 if the end of a directory listing is detected, at which point the print offer is also made. If the user makes a printout the arrays are re-set for the next page and the next directory. If no print is made the array must continue to fill without losing the filenames so far gathered. The FOR...NEXT loop therefore restarts not at 1 but at the present value of X, its identifier.

The innermost loop cycles once for each filename in the directory listing; it is here

at the centre of the nested iterations that the call to the wildcard-matching routine takes place. In all, this procedure may have exceeded Papert's ideal of a mind-sized segment of code but it presents in a single place the dynamics behind obtaining, filtering, storing and printing filenames. The mechanics of these processes, of course, lie in the subordinate definitions.

In comparison with *Search_Mode* the slaves which serve it are straightforward. Listing 17 obtains a directory listing, having first used Menu 4.1 to obtain the user's permission. the technique is simple – a channel is opened to a newly-created file and a directory listing diverted to that channel. The channel can then be closed and re-opened to read the file contents. Directory listings are headed by the name the medium was given when it was formatted, followed by the number of free sectors. These lines of information are read from the file and used to alter the program's information display. When control is passed back to the master procedure, therefore, the next line to be read from the file will be a filename.

Approval

Listing 18 also seeks the user's approval through a menu before carrying-out its task of printing a page of data. Each page of output is headed by the date and the search pattern being used. Each filled element of the *File\$* display is then printed, the last being followed by the Epson code for a linefeed.

The *File_Check* function at listing 19 contains only a menu offering facilities controlled by its three subordinate procedures. Its menu is displayed after each successful filename match if the user has selected the "Examine" option from the main menu. The default option is to continue the search but users can elect to place the file contents on to the screen or the printer or to delete the file. The menu is repeated so that users can scan a file on the screen, obtain a print of its contents and delete it with three consecutive menu selections.

Listing 20 controls the display of file contents on the screen. A new channel is

Listing 19

```

1900 DEFine FuNction  File_Check
1905 LOCal Loop,  key
1910 Menu 3
1915 REPeat Loop
1920 key = Bar_Menu (max)
1925 SElect ON key
1930 = 1: RETurn 0
1935 = 2: Show_File
1940 = 3: IF Delete_File: RETurn 0
1945 = 4: Print_File
1950 = 5, 0: RETurn 1
1955 END SElect
1960 Menu 3
1965 END REPeat Loop
1970 END DEFine File_Check

```


opened to the selected file and its contents are obtained line by line in a loop. This procedure is only slightly slower than using the COPY command and has the advantage of permitting users to stop the listing by pressing the ESCape key at any time. By selecting a menu option, users can continue the listing again or abandon it.

Listing 21 controls the deletion of files, an irrevocable step which, once taken, is often regretted. It is the responsibility of good software to warn users of such impending doom and to offer an opportunity for the decision to be reconsidered and perhaps rescinded. In this program, selecting "Delete" from the *File__Check* menu leads to an "Are you sure?" menu before the file is finally removed from the medium.

The remaining listings are more benign, supervising the progress of file contents to the printer. In a structure reminiscent of the *Search__Mode* procedure a number of loops are nested to control the action. The outer loop cycles once per page, the middle loop iterates once for each line drawn from the target file and the inner loop repeats itself each time a line is printed to the printer.

If the need for a new page is detected the *New__Page* function is called. It prints a header, increments the page counter and waits patiently until the user indicates that the printer is ready to go. If no more pages are required control returns to the *File__Check* menu.

The file management program is now complete. It is still fairly crude, particularly with regard to device handling, because of the limitations of SuperBasic and the need to compress the program to manageable length. Nevertheless it should prove to be a useful addition to your software library.

Listing 22

```
2200 DEFine PROCedure Print_File
2202 LOCAl Per_Page, Per_Input, Per_Line
2204 LOCAl PageNum, R$, Lyne
2205 PageNum = 1: OPEN#F, Dev$ & File$(X)
2207 REPEAT Per_Page
2208   IF NOT New_Page THEN EXIT Per_Page
2210   REPEAT Per_Input
2211     IF EOF(#F): EXIT Per_Input
2212     INPUT#F, R$
2213     REPEAT Per_Line
2214       IF Lyne > 50
2215         PRINT#P, CHR$(12)
2216         IF NOT New_Page THEN EXIT Per_Page
2217       END IF
2218       PRINT#P; "      "; R$(1 TO 70)
2219       Lyne = Lyne + 1
2220       IF LEN(R$) > 70
2221         R$ = R$(71 TO)
2222       ELSE
2223         EXIT Per_Line
2224       END IF
2225     END REPEAT Per_Line
2226   END REPEAT Per_Input
2227 END REPEAT Per_Page
2228 CLOSE#F
2229 END DEFine Print_File
```

Listing 23

```
2300 DEFine FuNction New_Page
2305 Menu 4.4
2310 IF Bar_Menu (2) = 2
2315   PRINT#P, File$(X)!!!"Page "; PageNum\\
2320   Lyne = 1: PageNum = PageNum + 1
2325   RETURN 1
2330 END IF
2335 RETURN 0
2399 END DEFine New_Page
```

Listing 20

```
2000 DEFine PROCedure Show_File
2005 LOCAl Loop, r$
2010 CLS: OPEN_IN#F, Dev$ & File$(X)
2015 REPEAT Loop
2020   IF INKEY$ = CHR$(27)
2025     Menu 4.5: IF Bar_Menu(2) = 1: EXIT Loop
2030   END IF
2035   IF EOF (#F): EXIT Loop
2040   INPUT#F, r$: PRINT r$
2045 END REPEAT Loop
2050 CLOSE#F: PRINT FILL$("^", 40)
2055 END DEFine Show_File
```

Listing 21

```
2100 DEFine FuNction Delete_File
2105 Menu 4.3
2110 IF Bar_Menu(2) = 2
2115   DELETE Dev$ & File$(X)
2120   PRINT "File deleted": RETURN 1
2125 END IF
2130 RETURN 0
2135 END DEFine Delete_File
```



Next month: Many programs can be enhanced by the judicious use of computer graphics. Mike Lloyd reveals some simple but effective graphics routines and also explains the trigonometrical commands found in SuperBasic.

JUPITER by Albert Arranz Cortes

This program is a simulator of the moon system of *Jupiter*. It works in real-time and it is fast enough to see the four moons revolving round the mother planet.

The program has some simple but

interesting features; you can choose a date and see the position of each satellite on that date. I went to ASTER, the astronomical society in Barcelona, to check my results and found only very small differences from its figures.

There is also an option to see the small moons revolving in an equatorial plane. You can also change the interval increment – normally one hour. The program could also be used for modelling the revolutions of the planets around our sun and I am working on that version.

```

1000 REMark *****
1010 REMark CALCULATION OF THE POSITION
1020 REMark OF THE GALILEAN SATELLITES
1030 REMark OF JUPITER
1040 REMark (c) Albert Arranz Cortes
1050 REMark BARCELONA 1989
1060 REMark *****
1070 :
1080 INIVAR
1090 INIMES
1100 PRESENTACIO
1110 PTEMPS
1120 CALCULA
1130 JUPITER1
1140 :
1150 DEFine PROCedure INIVAR
1160 DIM DM(12),DN(12)
1170 A=50:B=5:C=0:D=PI/180
1180 H=4:P=-1:Y=40
1190 FC=2:IT=1:T$="h"
1200 END DEFine INIVAR
1210 :
1220 DEFine PROCedure INIMES
1230 J=-1:L=0:DI=0
1240 RESTORE 1290
1250 FOR M=1 TO 12
1260 READ DM(M)
1270 DN(M)=DM(M)
1280 END FOR M
1290 DATA 31,28,31,30,31,30
1300 DATA 31,31,30,31,30,31
1310 END DEFine INIMES
1320 :
1330 DEFine PROCedure PRESENTACIO
1340 MODE 4:WINDOW 512,256,0,0
1350 PAPER 0:INK 7:CLS
1360 OVER 1
1370 OPEN#3,SCR_512X50A0X0
1380 OPEN#4,SCR_274X18A116X210
1390 OPEN#5,SCR_286X36A110X200
1400 OPEN#6,SCR_414X200A48X36
1410 BORDER#6,2,7
1420 INK 3
1430 FOR X=160 TO 175
1440 Y=Y+1
1450 IF X=175:INK 5
1460 CSIZE 2,1:CURSOR X,Y
1470 PRINT "J U P I T E R"
1480 PAUSE 1
1490 END FOR X
1500 OVER 0
1510 CSIZE 0,0:INK 7
1520 CURSOR 200,Y+23
1530 PRINT "and its satellites"
1540 CSIZE 1,0:INK 3
1550 CURSOR 185,180
1560 PRINT "by Albert Arranz"
1570 CURSOR 145,200
1580 PRINT "(c) Copyright 1989 Ver 3.01"
1590 CURSOR 193,218
1600 PRINT "- PRESS SPACE -"
1610 INK 4:FILL 1
1620 CIRCLE 73,50,3.5
1630 FILL 0
1640 INK 3:FILL 1
1650 CIRCLE 74.3,49,1.1,.3,1.9
1660 FILL 0
1670 REPEAT BUCLE
1680 X=COS(C*D)*A:Y=SIN(C*D)*B
1690 C=C+H
1700 INK 7:POINT X+73,Y+50
1710 K=CODE(INKEY$(0))
1720 IF K=32:GO TO 1110
1730 INK 0:POINT X+73,Y+50
1740 END REPEAT BUCLE
1750 CSIZE 1,0
1760 END DEFine PRESENTACIO
1770 :
1780 DEFine PROCedure PTEMPS
1790 CLS#6
1800 INK 4:UNDER 1
1810 CURSOR 210,40
1820 PRINT "ENTER DATE"
1830 CURSOR 164,53
1840 PRINT "IN U.T. (Univesal Time)"
1850 UNDER 0:INK 7
1860 CURSOR 100,80
1870 INPUT "1) Year in full (nnnn):"!ANY
1880 IF ANY MOD 4=0:DM(2)=29
1890 IF LEN (ANY)<>4
1900 INK 0:CURSOR 292,80:PRINT ANY
1910 INK 7:GO TO 1860
1920 END IF
1930 CURSOR 100,100
1940 INPUT "2) Month (1-12):"!MES
1950 IF MES>12 OR MES<1
1960 INK 0:CURSOR 235,100
1970 PRINT MES:INK 7:GO TO 1930
1980 END IF
1990 CURSOR 100,120
2000 INPUT "3) Day (1-31):"!DIA
2010 IF DIA<1 OR DIA>DM(MES)
2020 INK 0:CURSOR 220,120
2030 PRINT DIA:INK 7:GO TO 1990
2040 END IF
2050 DM(MES)=DIA+1
2060 CURSOR 100,140
2070 INPUT "4) Time (0-23):"!HOR
2080 IF HOR>23 OR HOR<0
2090 INK 0:CURSOR 226,140
2100 PRINT HOR:INK 7:GO TO 2060
2110 END IF
2120 CURSOR 204,190
2130 PRINT "Please wait"
2140 END DEFine PTEMPS
2150 :
2160 DEFine PROCedure CALCULA
2170 IF ANY=1985:GO TO 2320
2180 IF ANY>1985:GO TO 2260
2190 FOR AN=ANY TO 1984
2200 IF AN MOD 4=0
2210 ND=366:ELSE :ND=365
2220 END IF
2230 DI=DI-ND

```



```

2240 END FOR AN
2250 GO TO 2320
2260 FOR AN=1985 TO ANY-1
2270 IF AN MOD 4=0
2280 ND=366:ELSE :ND=365
2290 END IF
2300 DI=DI+ND
2310 END FOR AN
2320 FOR ME=1 TO 12
2330 IF ME>MES-1
2340 DI=DI+DIA:GO TO 2390
2350 END IF
2360 ND=DM(ME)
2370 DI=DI+ND
2380 END FOR ME
2390 HO=(DI*24)+HOR
2400 END DEFine CALCULA
2410 :
2420 DEFine PROCedure JUPITER1
2430 CLS
2440 INK 4:FILL 1
2450 CIRCLE 73,50,2.22
2460 FILL 0
2470 AI=16.22308:AE=25.82308
2480 AG=41.19231:AC=72.46154
2490 BI=AI*(-.17):BE=AE*(-.17)
2500 BG=AG*(-.17):BC=AC*(-.17)
2510 CAPSALERA
2520 INK 7
2530 CURSOR 170,65
2540 PRINT "JUPITER AND SATELLITES"
2550 INK 4
2560 LINE 1,32 TO 1,48
2570 LINE 32,32 TO 32,48
2580 LINE 89,32 TO 89,48
2590 LINE 98,32 TO 98,48
2600 INK 7
2610 CURSOR 0,175:PRINT "Callisto"
2620 CURSOR 108,175:PRINT "Ganymede"
2630 CURSOR 305,175:PRINT "Io"
2640 CURSOR 336,175:PRINT "Europa"
2650 LINE 124,0 TO 124,5
2660 LINE 124,5 TO 21,5
2670 LINE 21,5 TO 21,0
2680 LINE 21,0 TO 124,0
2690 INK 4
2700 CURSOR 90,245:PRINT "YEAR:"
2710 CURSOR 191,245:PRINT "MONTH:"
2720 CURSOR 281,245:PRINT "DAY:"
2730 CURSOR 357,245:PRINT "TIME:"
2740 INK 7
2750 REPEAT BUCLE
2760 I=(2*PI/42.47665)*(HO-979.85)
2770 VI=(HO-979.85)/42.47665
2780 I=I-(2*PI*INT(VI))
2790 E=(2*PI/85.29826)*(HO-1045.2)
2800 VE=(HO-1045.2)/85.29826
2810 E=E-(2*PI*INT(VE))
2820 G=(2*PI/171.9933)*(HO-1048.65)
2830 VG=(HO-1048.65)/171.9933
2840 G=G-(2*PI*INT(VG))
2850 C=(2*PI/402.0853)*(HO-1183.417)
2860 VC=(HO-1183.417)/402.0853
2870 C=C-(2*PI*INT(VC))
2880 INK 3
2890 CIRCLE 73,50,AI,.17,1.57
2900 CIRCLE 73,50,AE,.17,1.57
2910 CIRCLE 73,50,AG,.17,1.57
2920 CIRCLE 73,50,AC,.17,1.57
2930 INK 7
2940 CURSOR 133,245:PRINT ANY
2950 CURSOR 242,245:PRINT MES;" "
2960 CURSOR 316,245:PRINT DIA;" "
2970 CURSOR 400,245:PRINT HOR;" "
2980 XI=SIN(I)*AI:WI=73+XI
2990 XE=SIN(E)*AE:WE=73+XE
3000 XG=SIN(G)*AG:WG=73+XG
3010 XC=SIN(C)*AC:WC=73+XC
3020 YI=COS(I)*BI:ZI=50+YI
3030 YE=COS(E)*BE:ZE=50+YE
3040 YG=COS(G)*BG:ZG=50+YG
3050 YC=COS(C)*BC:ZC=50+YC
3060 POINT WI,ZI:POINT WE,ZE
3070 POINT WG,ZG:POINT WC,ZC
3080 IF J=1
3090 IF L=1:JUPITER2:L=0
3100 END IF
3110 K=CODE(INKEY$(P))
3120 SELEct ON K
3130 =9:INIMES:CLS:BORDER#6,2,7
3140 GO TO 1110
3150 =27:NEW
3160 =32:L=1:GO TO 3340
3170 =232:P=0:GO TO 3340
3180 =236:P=-1
3190 =240:J=1:J1=0:GO TO 3260
3200 =244:J=0:GO TO 3260
3210 =101,69:INCR
3220 CURSOR#3,245,20
3230 PRINT#3,T$
3240 GO TO 3110
3250 END SELEct
3260 IF J=1
3270 JUPITER2
3280 ELSE
3290 IF J=0
3300 CLS#5:BORDER#5,1,0
3310 END IF
3320 END IF
3330 IF P<>0:GO TO 3110
3340 IF IT=1
3350 HOR=HOR+1:HO=HO+1:MTEMPS
3360 END IF
3370 IF IT=2
3380 DIA=DIA+1:HO=HO+24:MTEMPS
3390 END IF
3400 INK 0
3410 POINT WI,ZI:POINT WE,ZE
3420 POINT WG,ZG:POINT WC,ZC
3430 END REPEAT BUCLE
3440 END DEFine JUPITER1
3450 :
3460 DEFine PROCedure JUPITER2
3470 BORDER#5,1,7
3480 IF J1=1:CLS#4:GO TO 3490
3490 INK 7:FILL 1
3500 CIRCLE 73,15,1.16
3510 FILL 0
3520 POINT (XI/FC)+73,15
3530 POINT (XE/FC)+73,15
3540 POINT (XG/FC)+73,15
3550 POINT (XC/FC)+73,15
3560 J1=1
3570 END DEFine JUPITER2
3580 :
3590 DEFine PROCedure CAPSALERA
3600 PAPER#3,0:INK#3,3
3610 BORDER#3,2,7:CLS#3
3620 CSIZE#3,0,0
3630 CURSOR#3,36,6
3640 PRINT#3,"«F1» Continued orbit |"
3650 CURSOR#3,183,6
3660 PRINT#3,"«F2» Stop continued orbit |"
3670 CURSOR#3,361,6
3680 PRINT#3,"«F3» Zero degrees"
3690 CURSOR#3,18,20
3700 PRINT#3,"«F4» Erase plane |"
3710 CURSOR#3,131,20
3720 PRINT#3,"«SPACE» Forward 1 |"
3730 CURSOR#3,268,20
3740 PRINT#3,"«TAB» New date |"
3750 CURSOR#3,368,20
3760 PRINT#3,"«ESC» Exit to BASIC"
3770 CURSOR#3,18,33
3780 PRINT#3,"«E» Interval increment"
3790 CURSOR#3,155,33
3800 PRINT#3,"change (1h / 1d) |"
3810 INK#3,7

```



```

3820 CURSOR#3,245,20:PRINT#3,T$
3830 END DEFine CAPSALERA
3840 :
3850 DEFine PROCedure INCR
3860 CURSOR#3,268,33
3870 PRINT#3,"«H» 1h | «D» 1d |"
3880 CURSOR#3,374,33
3890 PRINT#3,"«ESC» Abort comand"
3900 I$=CODE(INKEY$(-1))
3910 IF I$=72 OR I$=104
3920 IT=1:T$="h":GO TO 3990
3930 END IF
3940 IF I$=68 OR I$=100
3950 IT=2:T$="d":GO TO 3990
3960 END IF

```

```

3970 IF I$=27:GO TO 3990
3980 GO TO 3900
3990 CURSOR#3,268,33
4000 PRINT#3,"
4010 CURSOR#3,374,33
4020 PRINT#3,"
4030 END DEFine INCR
4040 :
4050 DEFine PROCedure MTEMPS
4060 IF HOR>23:HOR=0:DIA=DIA+1
4070 IF DIA>DN(MES)
4080 DIA=1:MES=MES+1
4090 END IF
4100 IF MES>12:MES=1:ANY=ANY+1
4110 END DEFine MTEMPS

```

PIP by Matthew Arends

One of the most annoying features of the QL is that there is no audio feedback from the machine when a key is pressed. When you are typing in a program, for instance, you have to keep glancing at the screen to see if the QL has responded. To combat the deficiency I wrote *Pip*.

Pip is an easy-to-use extension to the QL system. All that is needed is to type-in the SuperBasic program, RUN it and SAVE the machine code produced on to a Microdrive cartridge. To link-in the program, enter the following:
 p=RESPR(110)
 LBYTES mdv1_pip_bin, p
 CALL p

From then, every time a key is pressed – except CTRL, ALT and SHIFT on their own – the loudspeaker will respond with a

'pip'. For the sake of speed, when a key is held down and auto-repeats, only the initial 'pip' will be heard. Pip will not interrupt a sound already playing. Pip should be compatible with every QL program. As an example it is easy to use Pip with QL Quill on an expanded machine. To turn off pip, use the command:

CALL p+18

Pip can be restarted with CALL p and so on. The pitch and duration of the pip can be altered by changing the variables in the SuperBasic program before it is RUN or by issuing the following commands once it has been installed:

POKE p+90, pitch

and/or

POKE p+94, duration

Both must be byte-sized – smaller than 256. It is impossible to have a duration of longer than 255 units. To do this the original duration must be split into two numbers, lo and hi, where:

hi = duration VID 256 and
 lo = duration MOD 256

Then:

POKE_W p+94, lo*256+hi

To return to using byte-sized durations, remember to POKE p+95,0 first. For maximum ease of use, put the Pip loading routine in your Boot program.

```

mdv1_pip_boot 1988 Aug 10 21:13:
38 Wed

```

```

100 p=RESPR(110)
110 LBYTES mdv1_pip_bin,p
120 CALL p

```

```

mdv1_pip_prog 1988 Aug 10 21:12:
18 Wed

```

```

100 p=RESPR(110):tot=0:RESTORE
110 pitch=5:duration=100:REMark
both must be <256
120 FOR a=p TO p+98 STEP 2
130 READ b
140 POKE_W a,b
150 tot=tot+b
160 NEXT a
170 IF tot<>428018+pitch*256+duration*256:PRINT#0,'ERROR IN DATA
':STOP

```

```

180 INPUT#0,'Insert cartridge in
to mdv1 and press ENTER';a$
190 SBYTES mdv1_pip_bin,p,110
200 CALL p
210 DATA 17402,26,16890,94,8521,
4,28700
220 DATA 20033,20085,16890,80,28
701,20033
230 DATA 20085,28672,20033,12328
,140,21248
240 DATA -20376,144,26112,30,213
52,144
250 DATA 18426,24,28689,20033,20
49,1,26112
260 DATA 10,18426,16,28689,20033
,20085,256
270 DATA -21846,-21846,512,2568,
0,-21846
280 DATA pitch*256,0,duration*25
6,0,256

```


BINQL by Vic Newton

BinQL is a bingo-callers' utility. For the non-conversant, in the game of bingo numbers between one and 90 are called in random order and no number is repeated. Players have individual cards, each containing a different set of 15 of the 90 numbers. The game is controlled by the caller, who calls the numbers as they appear. The first player to mark all 15 numbers on a card is the winner.

The program is written for a domestic TV set, so select F2. Option 1 on the menu generates the numbers 1 to 90 in random order. The numbers are displayed at the bottom centre of the screen as each is generated. The next number is not generated until a key is pressed. This also transfers the previous number into its location on the grid. The 0 key must be pressed again to end the game and display the menu. If Option 2 is selected from the

menu all the 90 numbers are shown in the random order of the last game.

The kernel of the program is the interaction between a string <bag\$> and an array <s\$>. String <bag\$> starts the game containing the 90 numbers. The array <s\$> consists of 90 two-digit strings. The numbers are, in effect, transferred one at a time from the string to the array. It is an important feature of the program that at the end of the game <bag\$> is a null string, while the array contains all the numbers in the random order of the CALLOUT and of the 'completion of transfer' loop.

1450: random number n is generated from the set 1 to 90. The highest number is deleted every circuit of the loop.

1460: string <s\$> (count) is made equal to the nth in the queue in string <bag\$>.

1470: checks if the nth number is the last in the string. If so...

1480: ... it is removed.

1500: if it is not the last number, it is cut out and the resulting two pieces of the string are joined.

So the queue of numbers in <bag\$> is reduced by one every circuit and the length of the queue is always the same as the maximum value of n. No repetition is possible.

Long blocks have been used to produce the grid.

1380 CURSOR 28,6: are the pixel coordinates for PRINTing in the top left location - trial and error discovery.

1620 CURSOR 28+s\$(count, 1)*40, 6+s\$(count, 2)*16. Note the 28 and 6. The 40 and 16 are the STEPs used for the grid generation. s\$(count, 1) is the first digit of s\$(count) and s\$(count, 2) the second digit.

The PROCedure printcall can be omitted. It would anyway have to be modified to suit specific printers. It is used to study the 'randomness' of the numbers.

```

1000 REMark          BINQL by Vic Newton
1010 :
1020 REMark          A BINGO CALLERS UTILITY
1030 :
1040 :
1050 MODE 8:PAPER#1,1:CLS
1060 count=0:RANDOMISE:menu
1070 REMark .....
1080 DEFine PROCedure menu
1090   OPEN#4,scr_420x180a44x25
1100   BORDER #4,17,1
1110   PAPER#4,1:INK#4,7:CLS#4
1120   PRINT#4\\," M E N U"\\\\"ENTER 0 to EXI
T. "\\,"1 to START"\\,"2 for listing of"\\,"
the last call."\\,"3 for printout of"\\," the
last call."
1130   a$=INKEY$(-1)
1140   IF a$='0'THEN STOP
1150   IF a$='1'AND count<>0 THEN shakebag
1160   IF a$='1'AND count=0 THEN initialise
1170   IF a$='2'THEN listcall
1180   IF a$='3'THEN printcall
1190   menu
1200 END DEFine menu
1210 REMark .....
1220 DEFine PROCedure initialise
1230 LET backup$="01020304050607080910111213141516
17181920212223242526272829303132333435363738394041
42434445464748495051525354555657585960616263646566
676869707172737475767778798081828384858687888990":
bag$ = backup$:DIM s$(90,2)
1240 call_out
1250 END DEFine initialise
1260 REMark .....
1270 DEFine PROCedure call_out
1280 LOCAL loop,zero,a$,n,x,y,k,j
1290 REMark ... Set up the grid ...
1300 CLS:CLS#0: count=0

```



```

1310 FOR x=20 TO 440 STEP 40
1320   BLOCK 2,160,x,2,7
1330 NEXT x:END FOR x
1340 FOR y=2 TO 170 STEP 16
1350   BLOCK 400,2,20,y,7
1360 NEXT y:END FOR y
1370 REMark ... Mark unwanted locations ...
1380 CURSOR 28,6:PRINT "<>"
1390 FOR k=22 TO 150 STEP 16
1400   CURSOR 388,k:PRINT "<>"
1410 NEXT k:END FOR k
1420 REMark .. Start picking the numbers ..
1430 REPEAT loop
1440   count=count+1
1450   n = RND (1 TO 91-count)
1460   s$(count) = bag$(2*n-1 TO 2*n)
1470   IF LEN (bag$)<2*n+1 THEN
1480     bag$=bag$(1 TO 2*n-2)
1490   ELSE
1500     bag$= bag$(1 TO 2*n-2)&bag$(2*n+1 TO)
1510   END IF
1520   AT 17,1:PRINT count-1;" calls made"
1530   AT 17,20:PRINT "Last call No ";s$(count-1)
1540   AT 18,11:PRINT "CALL ";;FLASH 1:PRINT;s$(count);:FLASH 0:PRINT;" NEXT"
1550   AT 19,1:PRINT "Press a key to continue (0 to QUIT)"
1560   a$=INKEY$(-1)
1570   REMark .. The normal exit at end of game .
1580   IF a$='0' THEN
1590     AT 19,1:PRINT "
     ":EXIT loop
1600   END IF
1610   REMark .. Place number in its location ..
1620   CURSOR 28+s$(count,1)*40,6+s$(count,2)*16:
PRINT s$(count)
1630 :
1640   REMark ... Compulsory exit if all the
90 numbers are displayed in the grid ...
1650 :
1660   IF count=90 THEN
1670     AT 17,1:PRINT 90:AT 17,33:PRINT s$(count)
)
1680     AT 18,8:PRINT"All numbers displayed"
1690     AT 19,1:PRINT"          PRESS THE '0' KEY
"
1700     REPEAT zero
1710       a$=INKEY$(-1)
1720       IF a$='0' THEN EXIT zero:END IF
1730     END REPEAT zero
1740     EXIT loop
1750   END IF
1760 END REPEAT loop
1770 REMark .. Transfer all the remaining numbers
in bag$ to the array <s$> ..
1780 FOR j=count+1 TO 90
1790   n=RND (1 TO 91-j)
1800   s$(j)=bag$(2*n-1 TO 2*n)
1810   IF LEN (bag$)<2*n+1 THEN

```


PROCS

<				40	50				90
	11			41		61			<
02	12	22	32		52	62			<
03					53	63	73		<
04					54				<
05	15		35		55	65		85	<
06	16		36				76	86	<
07			37	47	57				<
08	18	28			58	68		88	<
09	19	29				69			<

43 calls made Last call No 54
CALL 56 NEXT
Press a key to continue (0 to QUIT)

22 35 50 16 88 08 53 65 69 32
63 40 62 41 18 58 15 55 76 06
86 36 85 09 02 05 07 90 73 37
12 52 19 11 28 57 61 29 03 68
04 47 54 56 83 42 49 87 84 44
38 45 23 39 33 26 81 30 71 17
31 72 10 89 25 67 27 70 13 64
14 01 79 78 59 46 75 74 80 51
60 48 43 21 34 24 82 66 20 77

Last call was 54 (Entry 43)

PRESS ANY KEY FOR MENU

<		20	30	40				80	
01	11	21	31	41	51		71	81	<
				42	52	62	72		<
		23		43	53	63	73	83	<
	14	24	34	44	54	64	74	84	<
	15	25	35		55		75		<
		26			56	66	76	86	<
07	17		37	47			77	87	<
08	18	28	38						<
09	19	29			59	69			<

55 calls made Last call No 62
CALL 12 NEXT
Press a key to continue (0 to QUIT)

01 18 29 31 47 40 30 66 44 42
53 52 09 43 80 26 77 81 63 37
19 28 21 87 56 38 25 55 71 86
69 15 24 07 35 17 34 74 14 76
51 75 59 64 11 08 54 23 41 72
20 84 83 73 62 12 04 67 02 46
05 88 60 27 06 13 49 78 58 22
79 16 70 57 10 50 65 90 61 89
45 39 48 32 36 33 68 85 82 03

Last call was 62 (Entry 55)

PRESS ANY KEY FOR MENU

```

1820      bag$=bag$(1 TO 2*n-2)
1830      ELSE
1840      bag$=bag$(1 TO 2*n-2)&bag$(2*n+1 TO )
1850      END IF
1860      NEXT j:END FOR j
1870      REMark      The bag is now empty ie bag$=""
1880      gameover
1890      END DEFine call_out
1900      REMark .....
1910      DEFine PROCedure gameover
1920      AT 19,1:PRINT "
          "
1930      CLS#0:PRINT#0,"  GAME OVER. Press any key f
or menu"
1940      PAUSE
1950      CLS#0:CLS:menu
1960      END DEFine gameover
1970      REMark .....
1980      DEFine PROCedure shakebag
1990      LOCAL j
2000      bag$=backup$
2010      REMark .. The 90 numbers now in <bag$> are
put in the random order of the last game ..
2020      FOR j=1 TO 90
2030      bag$(2*j-1 TO 2*j)=s$(j)
2040      NEXT j:END FOR j
2050      call_out
2060      END DEFine shakebag
2070      REMark .....
2080      DEFine PROCedure listcall
2090      LOCAL i
2100      CLS#4

```


PROGS

```

2110 IF count=0 THEN PRINT#4\\\" No call has be
en made\":PAUSE 100:menu:END IF
2120 FOR i=1 TO 90:PRINT#4;!s$(i)!
2130 END FOR i
2140 IF count=90 THEN LET count=91:END IF
2150 PRINT#4:PRINT#4; \" Last call was \";s$(count
-1);\" (Entry\"!count-1;\"))!
2160 PRINT#4\\\" \" PRESS ANY KEY FOR MENU\"
2170 PAUSE:menu
2180 END DEFine listcall
2190 REMark .....
2200 DEFine PROCedure printcall
2210 LOCAL i
2220 IF count=0 THEN CLS#4:PRINT#4\\\" No call
has been made\":PAUSE 100:menu:END IF
2230 OPEN#5,par
2240 PRINT#5;CHR$(27);\"E\";:REMark emphasized pic
a
2250 PRINT#5;CHR$(27);\"1\";CHR$(0);:REMark left m
argin at 0
2260 PRINT#5;CHR$(27);\"N\";CHR$(6);:REMark sets s
kip perforations (60 lines per page)
2270 FOR i=1 TO 90:PRINT#5;s$(i);' '
2280 END FOR i
2290 IF count=90 THEN LET count=91:END IF
2300 PRINT#5;' Last call was 's$(count-1);' (
Entry No 'count-1;')':PRINT#5
2310 CLOSE#5:menu
2320 END DEFine printcall
2330 REMark .....
END OF PROGRAM

```

TAKE YOUR PICK



Sinclair QL World.
Support for the
ever-popular QL. £1.75.



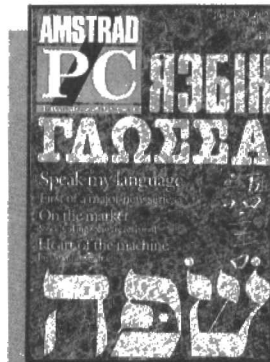
**Amstrad Computer
User.** Official title for
CPC Users. £1.25.



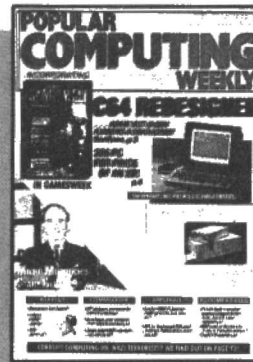
Which PC. The critical
PC Buyers' Guide.
£1.45.



Amstrad PCW. Official
title for Amstrad 8000
and 9000 series. £1.45.



Amstrad PC. Official
title for Professional users.
£1.50



**Popular Computing
Weekly.** News, views,
advice. PLUS *Computer
Gamesweek*, 70p.

FOCUS MAGAZINES, THERE'S ONE FOR EVERY COMPUTER USER

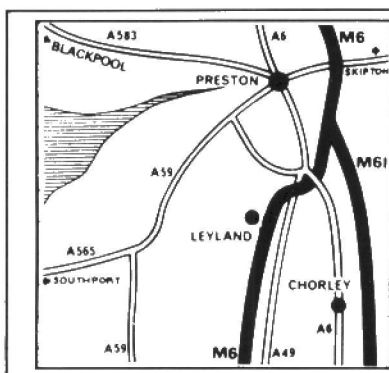
Order your copy from your
newsagent today. Subscriptions
available. Focus Magazines,
Greencoat House, Francis
Street, London SW1P 1DG.
Tel: 01-834 1717.

THE 3RD NORTHERN HOME COMPUTER SHOW

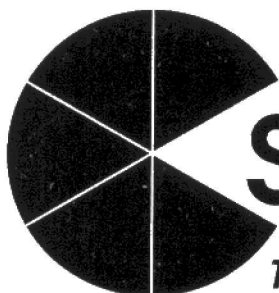
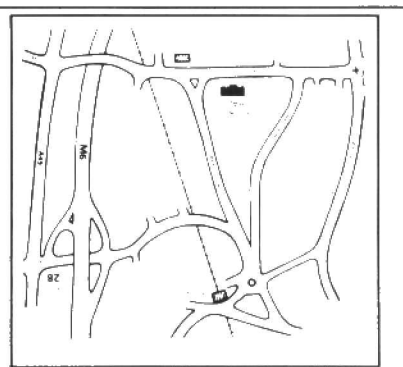
- *Over 35 stands for the home computer user*
- *Featuring all the major QL suppliers including Sector Software*
- *QL hardware, software, peripherals, printers, disk drives and games*
- *Restaurant and bar open all day*
- *Easy access by road and rail and bus*
- *Admission only £1.50*
- *The ONLY Christmas micro show for QL owners*

Don't miss the last chance to pick up your Christmas QL Goodies at the Third Northern Home Computer Show on Saturday 2nd December at Stokes Hall, Church Road, Leyland, Lancashire. Doors open 10am until 5pm.

For further information contact the organiser, David Batty on Leyland (0772) 454328



***HOW
TO GET
THERE***



ORGANISED BY –

Sector Software

The best programs and peripherals for the QL